الطريق من Action Script 2

الى Action Script 3

المــؤلــف

مبرمج

مـايـكـل نـبـيـل اخـنـوخ

Michaelnabil2006@yahoo.com

إهداء

اهدي هذا الكتاب بباقة ورد عطرة

الى صديقى العزيز/ شريف محمد الشاذلى

هو اول من علمنى حرفا فى برنامج الفلاش و لغة الاكشن اسكربت و له الفضل فى اهتمامى بهذا المجال.

و ايضا الى خطيبتى الغالية التى ساعدتنى فى تنسيقة و اعدادة و مراجعة الاكواد و تطبيقها .

ملحوظة :

المادة العلمية لهذاالكتاب قد تم مراجعتها و لكنة غير مسئول عن الاخطاء التى قد تحدث من سوء التطبيق او السهو و الخطأ مع مراعاة ان الكتاب قد تم تحريرة على برنامج الورد لذلك قد تجد مسافات زائدة فى الكود لذلك فى حالة أكتشاف أى اخطأ برجاء ارسال بريد الكترونى للكاتب توضح بة الخطأ و الصفحة .

جميع الاراء الموجودة فى هذا الكتاب هى اراء تعبر عن راى الكاتب الشخصى حتى لو لم توثق بمراجع .

الملاحق تعتبر خارج نطاق الكتاب و قد وضعت كمرجع للقارئ و ليست من صنع الكاتب لذلك ليس على الكاتب ادنى مسئولية من التطبيق الخاطئ لهذة المعلومات و قد وضعت اسماء المواقع التى تم اقتباس هذة الملاحق منها حفاظا لحقوق اصحابها و ليقوم القارئ بزيارتها .

أرسل ملاحظاتِك وتعليقاتِك وآراءك واقتراحاتِك وأسئلتَك على العنوانِ البريديِّ التالي:

Michaelnabil2006@yahoo.com

ويُفضّل جعل عنوان الرسالة "تعليق على الكتاب"، حتّى يمكنني تمييزها عن باقي الرسائل.

في الانتظار..

.

الـهـدف

أن الهدف من هذا الكتاب هو توفير العلم بطريقة رخيصة و غير مكلفة و ايضا تبادل المعلومات و الخبرات فى هذا المجال من خلال التعليق على الكتاب و ارسال التعليقات عبر الاميل للكاتب فيتم الرد عليها و نشرها فى الكتاب القادم أما الملاحق التى تم نقلها من موقع الموسوعة العربية للكمبيوتر و الانترنت فذلك لان الموضوعات التى يتناولها الكتاب لها علاقة وثيقة بهذة المقالات و لكنها خارج الهدف من الكتاب .
فالهدف من الكتاب هو تعليم Action script 2.0 من البداية و حتى الاحتراف بأسلوب بسيط و سهل و دون تعقيد و بعيدا عن اسلوب الكود المحفوظ لتنفيذ عمليات معينة لاننى للأسف وجدت نسبة كبيرة من كورسات Action script 2.0 تعتمد على هذة الطريقة و بعد انتهاء الكورس تجد الطالب لا يستطيع انشاء برامج خاصة به .
اما بالنسبة للغة Action script 3.0 فهى التطوير و لكن الاساس البرمجى واحد لذلك وجدت انة يجب ان تتعلم القديم و تتطور للحديث .

الـمـحـتـويـات

الفهرس

مـقـدمــةالـكـتــاب

هذا الكتاب موجة لمن لدية خبرة و لو قليلة ببرنـامج الفلاش كبرنـامج للجرافيك و الرسم و قد قمت فى بداية الكتاب بوضع فكرة سريعة عن برنـامج الفلاش بصفة عامة و لكن لن نشرح برنـامج الفلاش بالتفصيل بل فقط لغة الاكشن اسكربت ٢ .

أدوات برنـامج فلاش

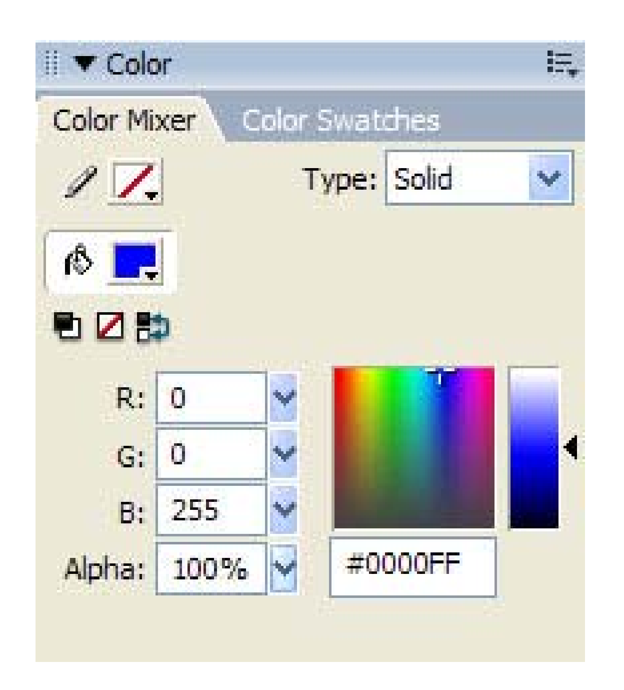
ستتعرف على معظم الأدوات المستخدمة في برنـامج فلاش .

لوحة الأدوات الرئيسية : تحتوي على أهم الأدوات المستخدمة في هذا البرنـامج ويحتوي هذا اللوح على الأدوات مثل أداة التحديد، أداة النص، أداة سطل التلوين، أداة رسم المستطيل، أداة رسم الدوائر، أداة الفرشاة، أداة القلم،أداة الممحاة ، أداة التحجيم ، أداة التكبير والتصغير .... ألخ ، سيتم شرح مهمة هذه الأدوات تدريجياً مع الدروس كما أنه يمكنك معرفة مهمة كل أداة من خلال وضع الماوس عليها ، الشكل رقم (١) يبين لوحة الأدوات الرئيسية .



الشكل (١)

لوحة التحكم بالالوان: **الوان العمل أو بألوان الأشكال أو الكتابات وهي مقسمة إلى قسمين ، القسم الأول لاختيار لون خط الرسم الثاني لاختيار لون الشكل أو التحكم في إظهار اللون من عدم إظهاره وغيرها ..... الشكل رقم (٢)**



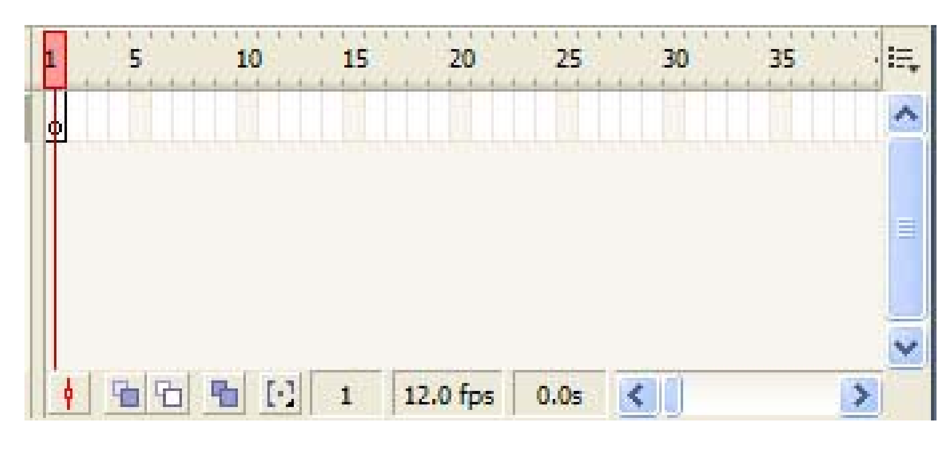
الشكل (٢)

نافذة الطبقات : وهي مهمة جدا ففيها يتم وضع الكائنات في طبقات محددة يتم التحكم بتغيير أسمها بالضغط نقرتين على نفس الطبقة وترتيبها وكذلك إخفائها أو حمايتها من التعديل أو حذفها ..... الشكل (٣) .



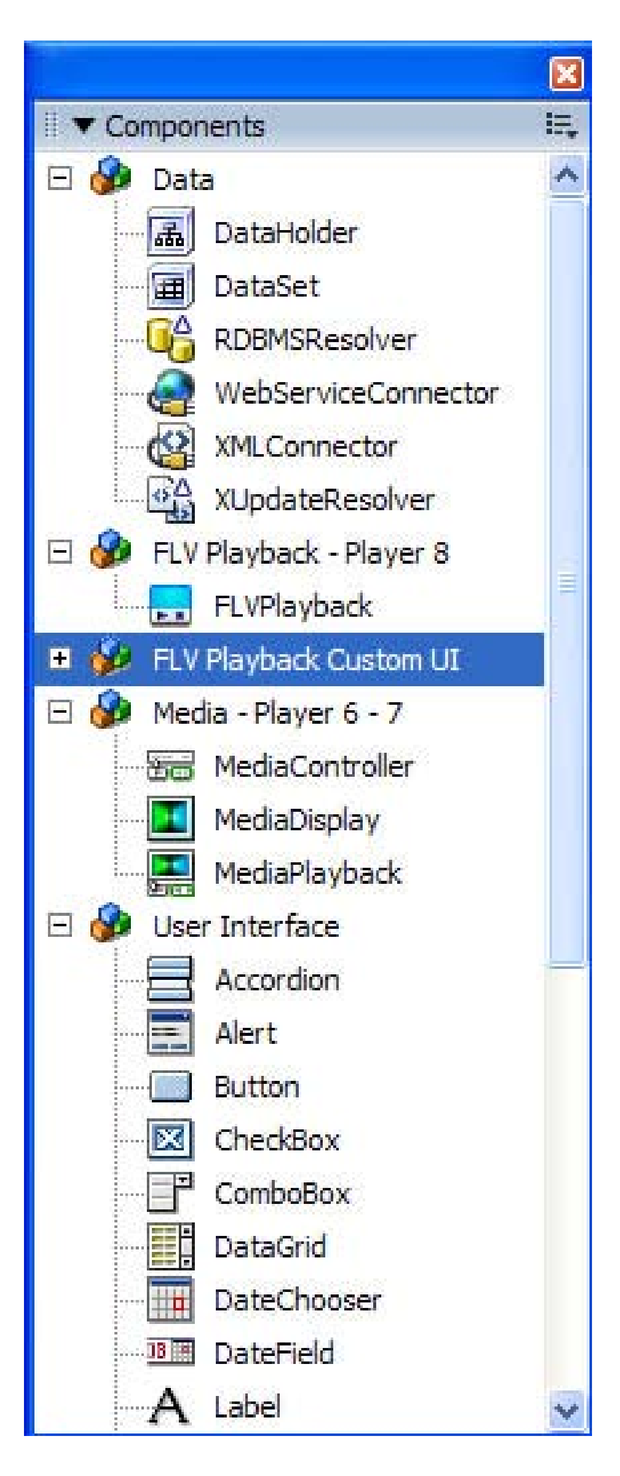
الشكل (٣)

شريط الوقت Time Line : حيث يقسم الوقت إلى إطارات Frames ، افتراضياً كل ١٢ إطار يمثل ثانية واحدة ، حيث يتم من خلاله تنسيق حركة الإطارات وتنظيم فيلم الفلاش ... الشكل (٤) .

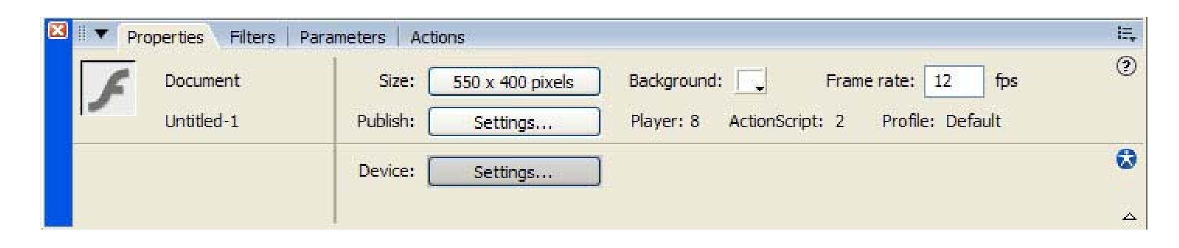


الشكل (٤)

لوح Components : **وهي تحوي العناصر الأكثر استخداماً في نظام ((الترابط التبادلي)) ، الشكل (٥) .**

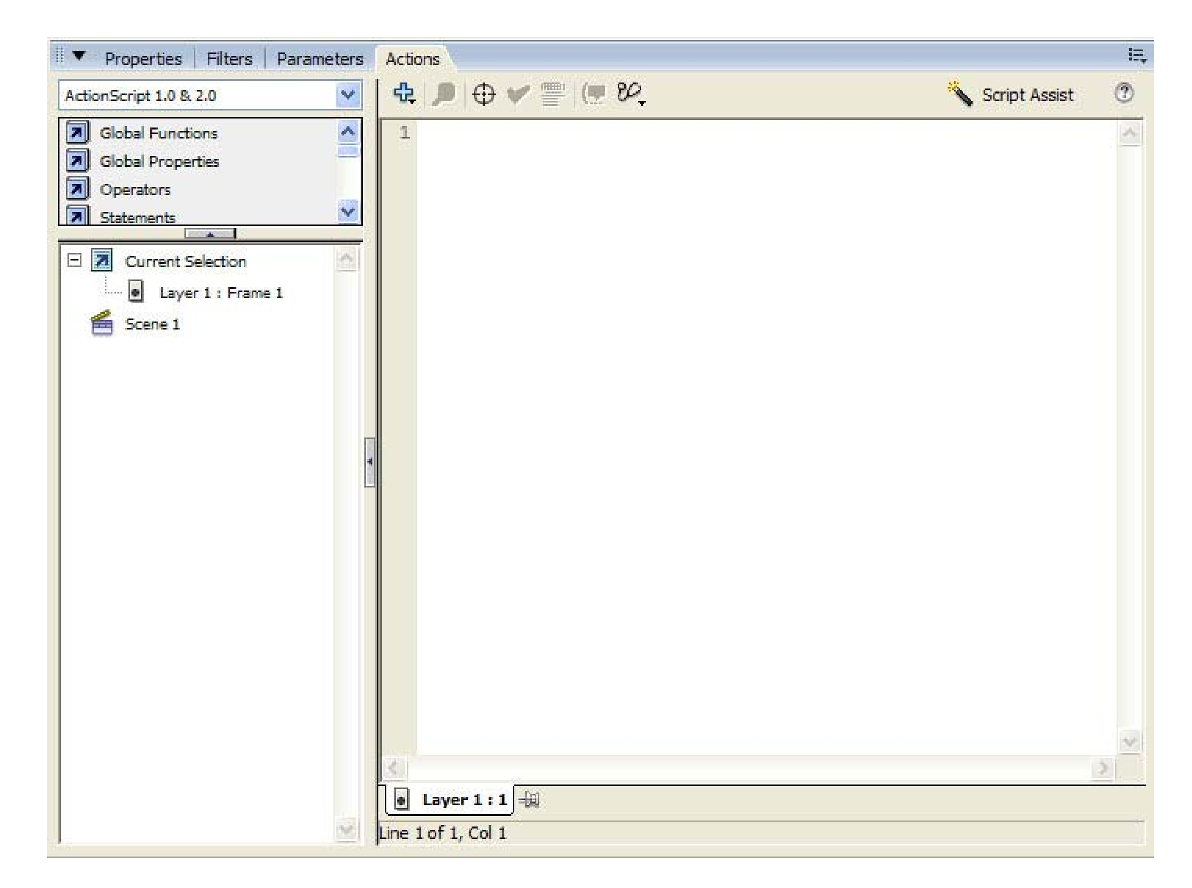


لوحة الخصائص Properties : وهي التي تتحكم بمعظم خصائص ملف فلاش والأشكال الموجودة ضمنه وستتعرف عليها خلال الدروس القادمة ، الشكل (٦) .



الشكل (٦)

لوحة الأحداث Actions : وهي تستخدم من أجل إضافة الأوامر البرمجية لفيلم الفلاش ، الشكل (٧) .



الشكل (٧)

الـكـتـابـة بـالـلـغـة العـربـيـة

من أهم مواضع القصور في برنامج فلاش للمستخدم العربي عدم دعمه للغة العربية ، بالتالي لا يمكن الاعتماد عليه وحده لتطبيق أدواته على نصوص باللغة العربية. إلا انه بالإمكان تجاوز هذا القصور عن طريق استخدام برنامج Corel Draw الذي يدعم اللغة العربية لإعداد النص المطلوب ومن ثم تصديره بصيغة ملف Adobe Illustrator إلى برنامج فلاش أو عن طريق برنامج الرسام العربي .

برنامج Corel Draw :-

١ - شغل برنامج الكوريل ثم استخدم أداة النص لكتابة العبارة المطلوبة يمكنك أيضا اختيار حجم الخط ولونه واسمه حسب الرغبة .

٢ - الآن سنستخدم أداة خاصة في Corel Draw لتحويل النص إلى انحناءات ، بعد تظليل العبارة التي كتبتها أختر Convert To Curves<Arrange من القائمة الرئيسية ، ستلاحظ أن تحويل النص إلى انحناءات قد قسم وباعد بين الحروف ، لا تهتم لذلك سنصلح هذا الخلل ضمن فلاش .

٣ - بعد ذلك من القائمة الرئيسية File أختر Export لتصدير النص ومن ثم ستحصل على مربع حواري يطلب منك تسمية الملف المراد تصديره واختيار نوعية التصدير ، أكتب اسما للملف ثم اختار Adobe Illustrator و اختصاره AI من القائمة المنسدلة ثم اضغط Export ثم OK ، هذا كل ما عليك فعله في الكوريل ...

٤ - اذهب الآن إلى برنامج فلاش وقم باستيراد النص الذي أعددناه في الخطوات السابقة من القائمة الرئيسية لبرنامج فلاش اختر Import من قائمة File ثم حدد الملف الذي حفظته .

٥ - لاحظ أن النص يظهر مقطع، من أجل تصليح هذا الأمر من القائمة الرئيسية **Modify** أختر **Break Apart** ، اضغط مفتاح **Shift** في لوحة المفاتيح ثم استخدم الزر الأيسر للماوس وانقر مرة واحدة على أي حرف ثم استخدم السهم الأيمن في لوحة المفاتيح وقم بإزاحة بقية النص إلى الجهة اليمنى حتى تلتصق الحروف مع هذا الحرف ، كرر نفس الطريقة مع بقية الأجزاء المقسمة الأخرى حتى تحصل على نص متكامل بنوعية جيدة .

** يمكنك الآن تغير لون الخط ومقياسه حسب الرغبة ، فقط اضغط **Ctrl+A** في لوحة المفاتيح ثم استخدام أداة التلوين لتغير لون الخط وأداة القياس لتحديد المقياس .

ﰡﰢﰣﰤﰡﰢﰣﰤﰡﰢﰣﰤ

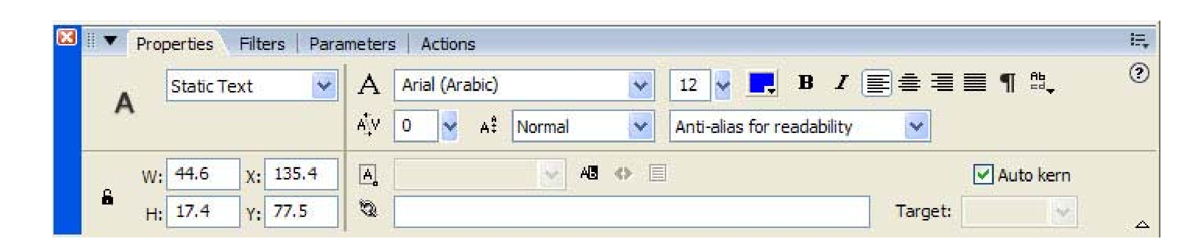
برنامج الرسام العربي :

١ - اكتب النص الذي تريد في واجهة برنامج الرسام العربي الرئيسية .

٢ - من القائمة الرئيسية **Edit** اختر **Copy & Minimize** سيحفظ جهاز الكمبيوتر نسخة من النص الذي كتبته في الخطوة الأولى ضمن الذاكرة .

٣ - عد الآن إلى برنامج فلاش استخدم أداة النص وانقر بها حيث المكان الذي ترغب في إضافة النص فيه ، الآن استخدم لوحة المفاتيح واضغط على **Ctrl+V** لإلصاق النص المحفوظ في الذاكرة ، وطالما إن برنامج فلاش لا يدعم اللغة العربية فستلاحظ ظهور النص على شكل رموز غير مقروءة .

٤ - من لوح الخصائص اختر القائمة المنسدلة التي تحتوي على قائمة بالخطوط **(Fonts)**الموجودة في جهازك ، ستجد مجموعة من الخطوط تبدأ بالأحرف **AXt** الإنجليزية . هذه هي الخطوط الخاصة ببرنامج الرسام العربي أختر الخط الذي تريد ولاحظ بأنك حصلت على النص العربي المقروء ، طبق الآن ما تريده من حجم الخط ولونه وذلك من خلال لوح الخصائص الشكل (١) .



الشكل (١)

وبذلك تكون قد عرفت كيف تكتب بـاللغة العربية في برنامج فلاش .

ﰡﰢﰣﰤﰡﰢﰣﰤﰡﰢﰣﰤ

إنشاء الحركة في فلاش ٨

سنتعلم من هذا الدرس كيفية كيفية تحريك الأشكال أو الصور من نقطة في الفيلم إلى نقطة أخرى حيث أنه هناك طريقتين لإنشاء الحركة في فلاش وهما (طريقة الحركة البينية للأطرات، طريقة إطار تلو إطار) وسوف نقوم بشرح كلتا الطريقتين مع الأمثلة المناسبة .

قبل أن تبدأ : عليك أن تتعلم كيف تخصص فيلم الفلاش من حيث أبعاده ولون الخلفية وعدد الإطارات في الثانية وما الى ذلك .

من القائمة الرئيسية Modify اختر Document أو اضغط على Ctrl + M يظهر لك مربع حوار Document Properties ومن ثم تقوم بتخصيص الفيلم كما تشاء كما هو موضح بالشكل (أ) :-

Document Properties

Title:

Description:

Dimensions: 550 px (width) x 400 px (height)

Match: Printer Contents Default

Background color:

Frame rate: 12 fps

Ruler units: Pixels

Make Default OK Cancel

Width: العرض ، height الطول ، Background color لون الخلفية ، frame rate : عدد الاطارات بالثانية، ruler units وحدة القياس .

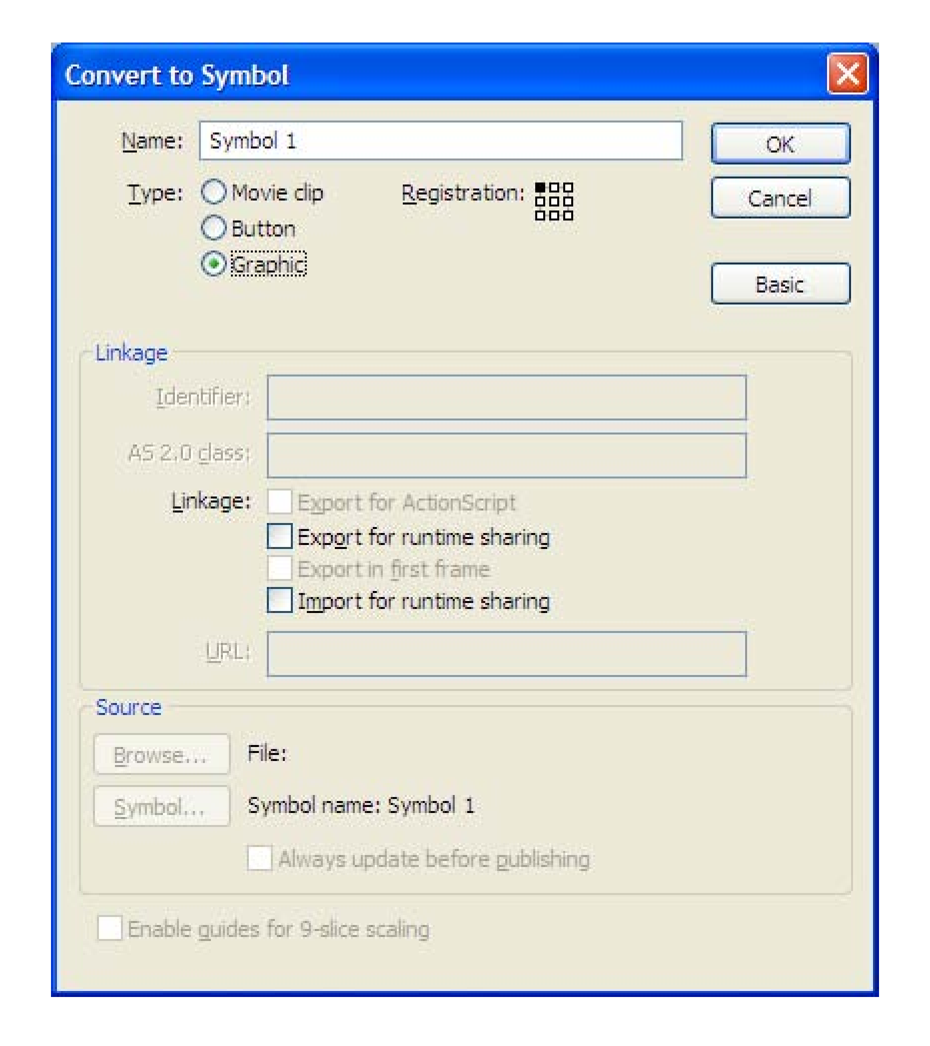
الشكل (أ)

أولا- طريقة الحركة البينية للأطر :

على فرض أننا سوف نقوم بتحريك دائرة من أول المشهد إلى آخره فتكون الخطوات :

١. اختر أداة الدائرة من شريط الأدوات ثم بعد تحديد لون التعبئة والخط الخارجي نقوم برسم الدائرة في الجانب الأيسر من المشهد، لاحظ أنك حصلت على دائرة سوداء في شريط الزمن عند الإطار رقم١ .

٢. قم بالنقر المزدوج على الدائرة من أجل تحديد لون التعبئة والخط الخارجي ثم نحولها إلى رمز من خلال الأمر Convert to Symbol من قائمة Insert أو من خلال الضغط على زر F8 فيظهر مربع الحوار الشكل(١) نحدد الخيار Graphic ثم نضغط OK .



الشكل (١)

بعد تحويل الشكل إلى رمز يقوم فلاش بحفظ الرسم في المكتبة الخاصة للمشروع يمكن رؤية ذلك من خلال الأمر Library من قائمة Window ((سنقوم بشرح التعامل مع المكتبات في الدروس القادمة)) .

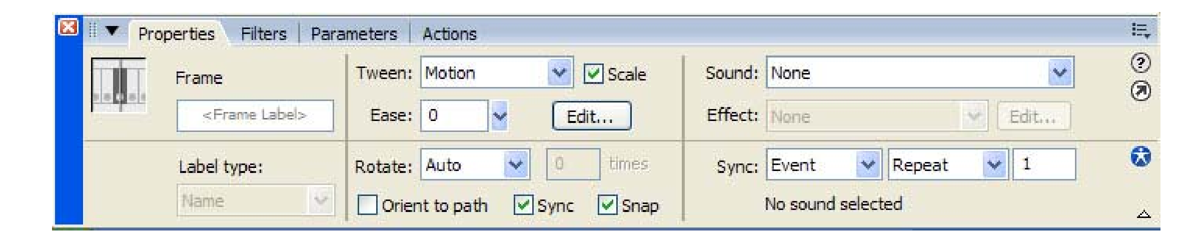
٣. الآن باستخدام الماوس نقوم بتحديد الإطار ٢٠ من شريط الزمن لنفس الطبقة بالزر الأيمن وتختار من القائمة السريعة الأمر Insert Keyframe أو نضغط الزرF6 حيث سيصبح شريط الزمن هكذا

ثم تقوم بتحريك الدائرة في مربع الرسم إلى الجهة المقابلة باستخدام الماوس .

٤. والآن سوف نقوم بإضافة الحركة إلى الفيلم وذلك باستخدام الماوس بالزر الأيمن نحدد الإطار الأول في شريط الزمن ونختار من القائمة السريعة الأمر Tween Create Motion فيصبح شريط الزمن على شكل سهم له خلفية زرقاء باهته مع العلم أنه لا تعمل الحركة البينية إلا مع الرموز أو الكائنات المجمعة .

٥. الآن نقوم باختبار ما قمنا به من خلال الضغط على زري Ctrl+Enter يجب أن تكون الحركة كما في الشكل (٢) :

الشكل (٢)



الشكل (٣)

يبين الاختيار Tween خيارات الحركة البينية حيث يمكن تحديد معدل الدوران حسب اتجاه عقارب الساعة أو بالعكس أيضا يمكن إلغاء الحركة البينية ويمكن أيضا تحديد خيارات السهول لسرعة الانطلاق أو سرعة الوصول .

ثانياً- طريقة الإطار تلو الإطار:

تعتمد هذه الحركة على تغير محتويات صفحة العرض في كل إطار وهي مناسبة للأشكال المعقدة (مع العلم أن طريقة الإطار تلو الإطار تحتاج لحجم ملف أكبر من الحركة البينية) .

على فرض أننا سنقوم بإنشاء ملعب لكرة السلة وفكرة المثال هي انطلاق الكرة ودخولها في الدائرة كما ترون إن هذا الشكل معقد بالنسبة للحركة الأولى ولقد استخدمنا طريقة الإطار تلو إطار بعد رسم الملعب طبعاً حيث عند كل إطار نقوم بإضافة إطار مفتاحي ثم نحرك الكرة قليلاً وهكذا حتى دخولها الدائرة وسقوطها على الأرض (ولا ننسى تحويل الكرة إلى رمز) .

التغير البيني للأشكال (تأثير المورفنج)

سنقوم بهذا الدرس بتعلم كيفية تغير الأشكال إلى أشكال أخرى تختلف في بنيتها عن طريق التداخل في بنيتها وسوف نقوم بشرح طريقتين لهذا التغير الأولى بالطريقة العادية والثانية باستخدام نقاط التحويل وهي مفيدة جداً من أجل ضبط التغير والتحكم به وسوف نشاهد ذلك من خلال المثالين التالين عن كل طريقة بحيث يشرح كل مثال كيفية التحول بين الأعداد ١و٢و٣:

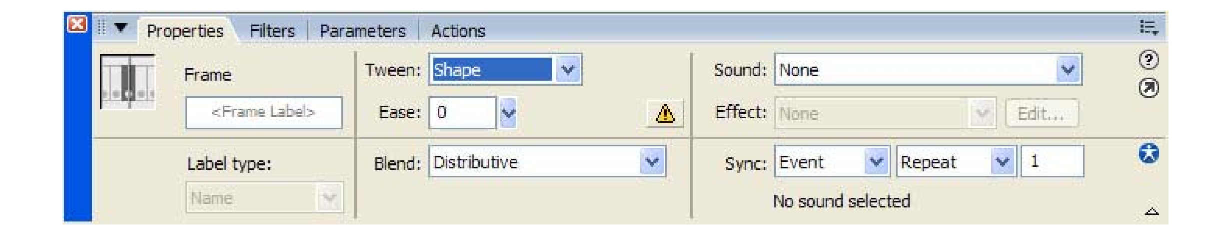
أولاً- الطريقة العادية:

١. نختار أداة النص ونقوم بكتابة الرقم١ في منطقة الرسم ثم نقوم بتكبيره ليشغل ثلثي منطقة الرسم من خلال الأداة عن طريق سحب مقابض التحجيم مع استمرار الضغط على مفتاح Shift من أن يكون التحجيم متوازي والمحافظة على الشكل، ثم نقوم بكسر العناصر من خلال الأمر Break Apart من قائمة Modify.

٢. عند الإطار١٥ ننقر عليه بالماوس ثم نضغط F6 لإضافة إطار مفتاحي ثم نضغط الزر Delete لحذف الرقم١ ثم نقوم بكتابة الرقم٢ ونجري عليه نفس الخطوة السابقة وكذلك الرقم٣ عند الإطار٣٠.

٣. الآن وبعد كتابة الأرقام الثلاثة وكسر عناصرها سوف يصبح شريط الزمن عندك هكذا

سوف نقوم الآن بإجراء التغير البيني للأرقام الثلاثة، اختر الإطار رقم واحد من شريط الزمن ومن لوح Properties نختار الحركة من نوع Shape من القائمة Tween كما في الشكل رقم (١).



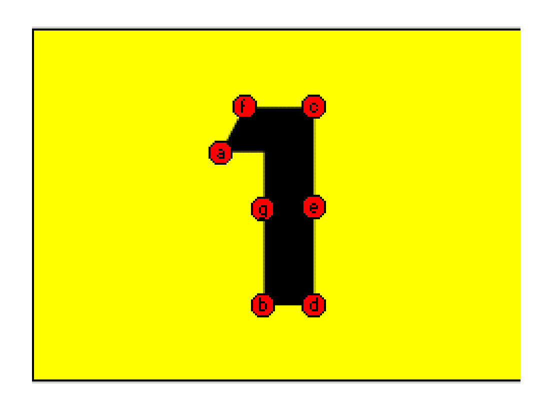
الشكل رقم (١)

سوف تلاحظ أن شريط الزمن قد تحول إلى سهم لونه أخضر فاتح، ونكرر نفس الخطوة عند الإطار رقم١٥ فيصبح شريط الزمن عندك هكذا الآن نقوم باختبار الفيلم بالضغط على الزر **Ctrl+Enter**

لاحظنا في المثال السابق كيف أن هناك تشويه وعدم تركيز أثناء التغير البيني للأرقام الثلاثة وبالتالي هنا يأتي دور نقاط التحويل للتحكم بتلك التغيرات.

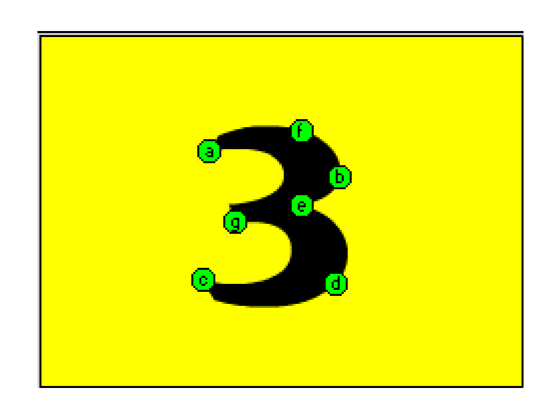
ثانياً- طريقة نقاط التحول:

بالعودة إلى المثال السابق عند آخر نقطة وصلنا لها نقوم بإضافة سبعة نقاط تحول عند الإطار ١ من خلال الأمر القائمة **Modify** القائمة الفرعية **Shape** الخيار **Hint Add Shape** نفس الخيار لكل نقطة أو من خلال الضغط على **Ctrl+Shift+H**، فيظهر عندنا سبعة نقاط لونها أحمر فوق بعضها البعض نقوم بواسطة الماوس بسحبها إلى زوايا مختلفة كما في الشكل رقم (٣):



الشكل (٣)

الآن نذهب إلى الإطار رقم١٥ ونقوم بسحب النقاط إلى زوايا مختلفة وعند إلغاء التحديد نلاحظ أنه قد تحول لون النقاط إلى الأخضر والنقاط ضمن الإطار رقم١ إلى اللون الأصفر ((دائماً يكون لون النقاط صفراء في إطار البداية وخضراء في إطار النهاية)) ولإظهار نقاط التحويل في حال عدم ظهورها نفعّل الخيار **Hints Show Shape** من القائمة **View**، ونكرر نفس الخطوة السابقة عند الإطار١٥ والإطار٣٠ كما في الشكل رقم (٤):

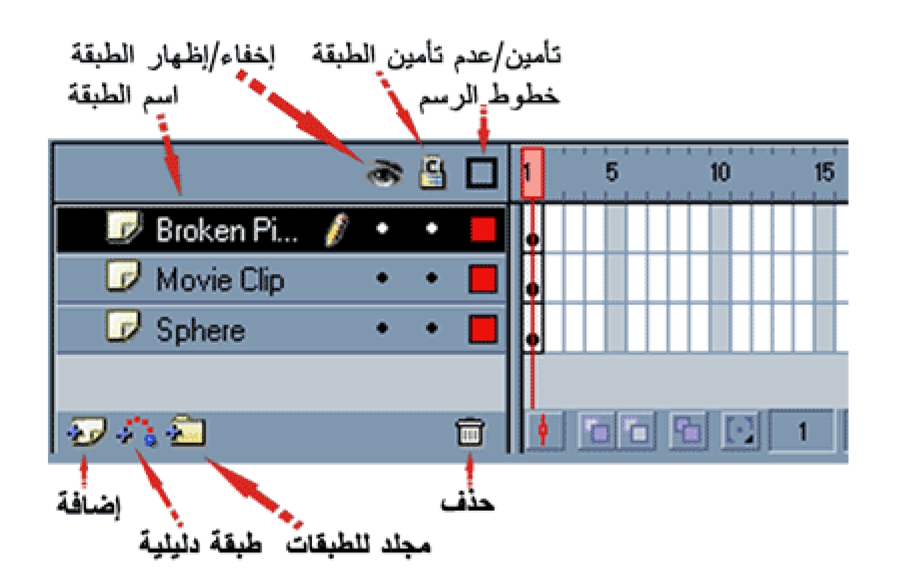


الشكل رقم (٤)

الآن نقوم بتنفيذ العمل بالضغط على الزر **Ctrl+Enter.**

الطبقات في فلاش ٨

إن استخدام الطبقات مهم جداً في فلاش لتنظيم العمل الفني ، وعندما تقوم بتحرير العناصر على الطبقة ، فإن عمليات التحرير التي تجريها لا تؤثر على العناصر الموجودة على الطبقات الأخرى ، كل طبقة تنشئها لها شريط الوقت الخاص بها كما في شكل رقم (١):



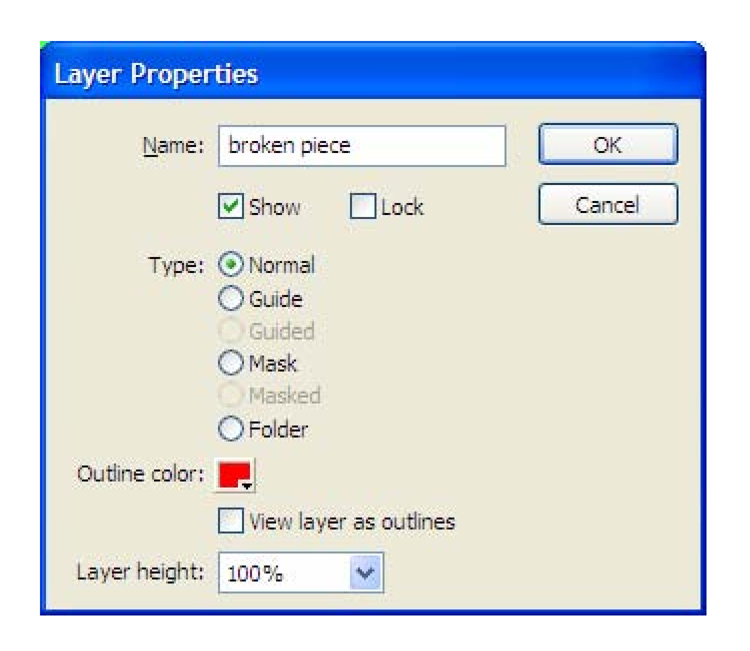
الشكل رقم (١)

أنواع الطبقات:

هناك أربعة أنواع للطبقات وهي: (الطبقة العادية ـ الطبقة القناع ـ الطبقة الدليلية ـ طبقة المجلد) حيث سنقوم بشرحها بالتفصيل مع الأمثلة إن لزم الأمر:

أولاً ـ الطبقة العادية:

تفيد تلك الطبقة فقط في فصل العناصر عن بعضها فعند رسم عنصرين متداخلين على نفس الطبقة يصبحان شكل واحد لذلك يفضل عند رسم أي عنصر أن يتم رسمه ضمن طبقة منفصلة، ويمكن تغيير اسم أي طبقة عن طريق النقر المزدوج على الاسم، ويمكن أيضا تعديل خصائص تلك الطبقة أو تحويلها لطبقة دليلية أو قناع من خلال النقر بالزر الأيمن على الطبقة واختيار الأمر **Properties** الشكل رقم (٢):



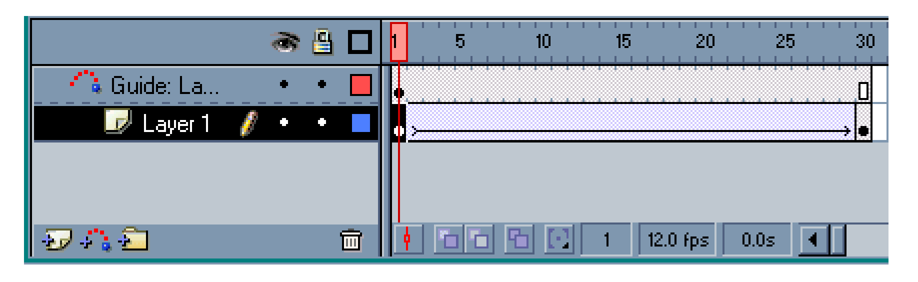
الشكل رقم (٢)

ثانياً- الطبقة الدليلية:

نقوم بإنشاء الطبقات الدليلية لاستخدامها كمرجع عند محاذاة وضبط مواضع العناصر ضمن الأفلام وطبقات دليل الحركة لإنشاء المسار الذي تتبعه العناصر أثناء حركتها ((مع العلم أن كل ما يرسم أو يكتب ضمن الطبقة الدليلية لا يظهر عند عرض الفيلم))، وكمثال على ذلك نفترض أننا سنقوم برسم كرة وتحريكها بمسارات عشوائية فتكون الخطوات:-

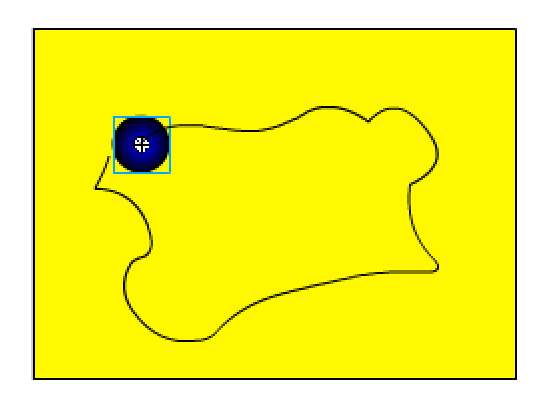
١. نقوم برسم الكرة ونحددها بالماوس بالنقر المزدوج لتحديد لون الملء والإطار معاً ثم نحولها إلى رمز من خلال الضغط على الزر F8 واختيار **Graphic**.

٢. ندرج إطار مفتاحي عند الإطار ٣٠ نحدد طبقة العمل ثم نضغط على زر إضافة طبقة دليلية يجب أن يكون شريط الزمن والطبقات مثل الشكل رقم (٣):



الشكل رقم (٣)

٣. نحدد الإطار رقم١ من الطبقة الدليلية وباستخدام قلم الرصاص نقوم برسم خط لا على التعين حيث سيكون ذلك الخط هو المسار الذي ستسير عليه الكرة أثناء العرض وليكن الخط كما في الشكل رقم (٤):



الشكل رقم (٤)

٤. نقوم الآن بتفعيل ميزة القفز إلى الدلائل اختر أمر دلائل الرسم **guides** من قائمة أمر العرض **view** ، ثم اختر أمر القفز إلى الدلائل **snap to guides**، ننتقل الآن إلى الطبقة الرئيسية ونحدد الإطار رقم١ ثم نقوم بسحب الكرة إلى رأس الخيط وعند الإطار رقم٣٠ نسحب الكرة إلى رأس الخيط المقابل بحيث يجب في الحالتين أن تظهر قرب الماوس دائرة سوداء تدل على ارتباط الدائرة بالخيط، الآن نضيف الحركة من خلال النقر بزر الماوس الأيمن واختيار الأمر **Create Motion Tween**، ثم تنفيذ الفيلم بالضغط على **Ctrl+Enter** .

ثالثاً- طبقة القناع:

هذا النوع من الطبقات مفيد جداً لعمل الخدع والأعمال الفنية حيث يتم وضع طبقة القناع فوق طبقة أو عدة طبقات أخرى لإخفاء أو إظهار أجزاء معينة من تلك الطبقة، فالمناطق المملوءة من طبقة القناع هي فقط التي ستظهر من الطبقة المقنعة، وتكون دائماً طبقة القناع فوق الطبقة المقنعة، وسنقوم بشرح مثال بسيط عن ذلك خطواته هي:

١. نحتاج إلى طبقتين الأولى للكتابة عليها والثانية لجعلها قناع، نقوم بإضافة طبقة ثانية من خلال الضغط على زر إضافة طبقة الأولى نسميها الكرة بالنقر المزدوج على اسمها والثانية نسميها كتابة.

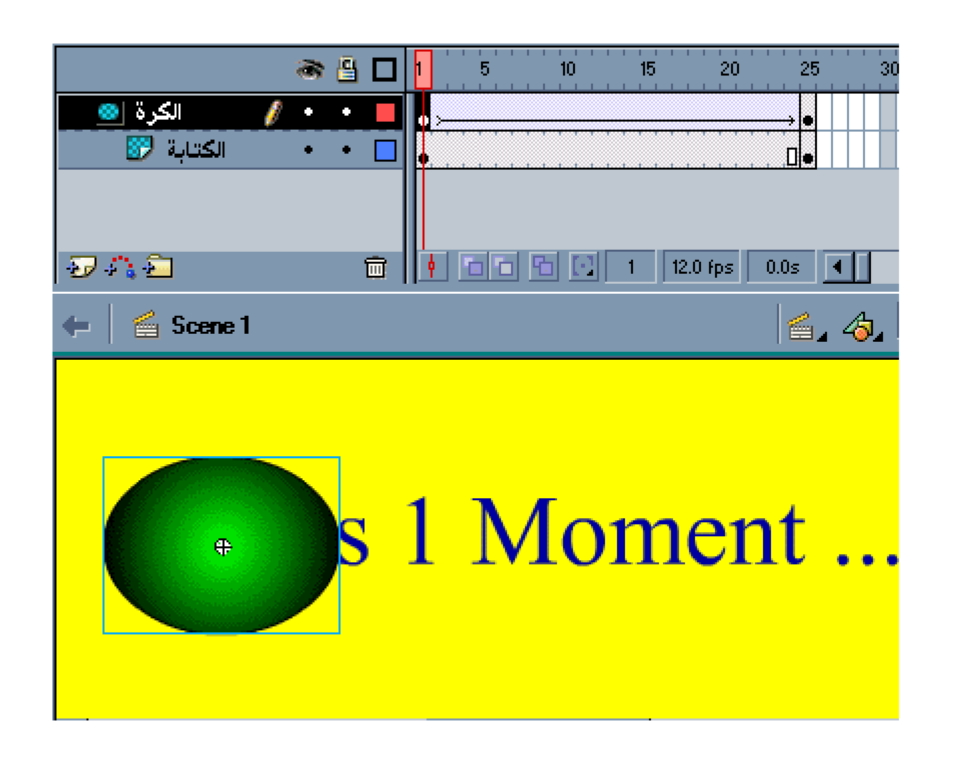
٢. نختار طبقة الكتابة وعند الإطار١ باختيار أداة الكتابة نكتب النص وليكن **One Moment Please!** ونكبره حتى يشغل معظم مساحة الرسم ثم نحوله إلى رمز بالضغط على زر **F8** من نوع **Graphic** ثم نقوم بإضافة إطار مفتاحي عند الإطار٢٥ بالضغط على الزر**F6**.

٣. الآن ننتقل إلى الطبقة الكرة نحددها بالنقر عليها وعند الإطار١ نقوم برسم الكرة عند بداية النص حيث لا يهم لون التعبئة لأنه لا يظهر عند العرض كما ذكرنا سابقاً، ثم نحول الكرة إلى رمز بالضغط على زر **F8** من نوع

Graphic ثم نقوم بإضافة إطار مفتاحي عند الإطار ٢٥ بالضغط على الزرF6 نحرك الكرة بواسطة الماوس إلى نهاية النص.

٤. نقوم الآن بإضافة الحركة إلى الكرة حيث نضغط على الإطار الأول من طبقة الكرة بالزر الأيمن للماوس ونختار الأمرCraete Motion Tween .

٥. الآن نقوم بإضافة القناع لطبقة الكرة بالضغط عليها بزر الماوس الأيمن واختيار الأمر Mask وبذلك يتغير رمز الطبقة كما في الشكل رقم (١):



الشكل رقم (١)

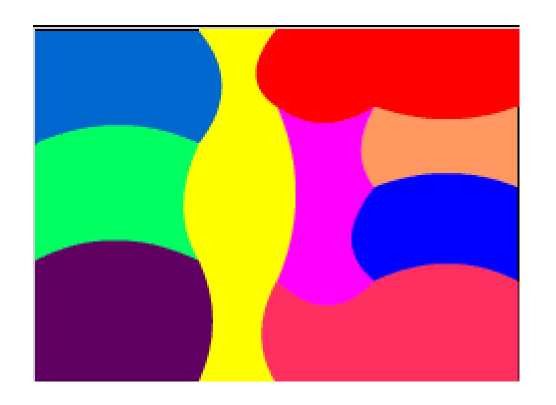
٦. الآن نقوم باختبار الفيلم بالضغط على Ctrl+Enter .

ملحوظة:- يمكن معاينة المشهد من شاشة الرسم من خلال تأمين طبقة القناع والطبقة المقنعة والتنفيذ بالضغط على زر Enter.

- سنقوم الآن بشرح مثال آخر عن تطبيقات طبقة القناع نظراً لأهميتها البالغة في جميع الأعمال الفنية، وفكرة مثالنا هي بسيطة جداً حيث تدور حول فكرة كتابة نص وتكبيره وبالتالي تغير الألوان ضمنه باستمرار وخطوات الدرس هي:

١. ننشئ طبقتين الأولى طبقة قناع Mask نسميها Text والثانية مقنعة Masked نسميها Colors .

٢. نختار طبقة Colors ونقوم برسم مستطيل ونلونه بألوان عشوائية حسب الرغبة مثلاً كما في الشكل رقم (٣) ونضيف إطار مفتاحي لنفس الطبقة عند الإطار ٤٥ بالضغط على مفتاح F6 ثم نقوم بتأمين تلك الطبقة لسلامتها:

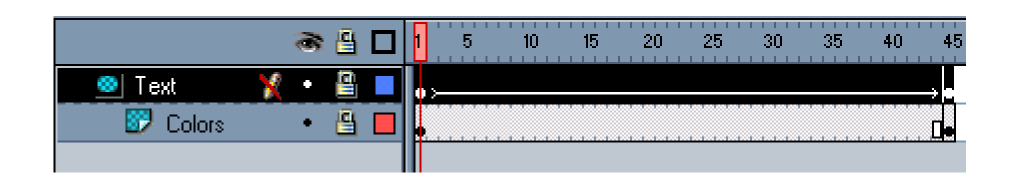


الشكل رقم (٣)

٣. الآن نختار الطبقة **Text** ونقوم بكتابة نص وليكن **COLOR** ثم نحوله إلى رمز بالضغط على مفتاح **F8** من نوع **Graphic** ونسحبها بواسطة الماوس إلى فوق نافذة الرسم، ثم نضيف إطار مفتاحي أيضا عند الإطار٤٥ بالضغط على مفتاح **F6** ثم نسحب النص إلى أسفل نافذة الرسم.

٤. نقوم بإضافة الحركة للنص عن طريق النقر بالماوس بالزر الأيمن على الإطار١ لطبقة **Text** ونختار الأمر **Create Motion Tween** .

٥. الآن نقوم بإضافة القناع لطبقة **Text** بالضغط عليها بزر الماوس الأيمن واختيار الأمر **Mask** وبذلك يتغير رمز الطبقة كما في الشكل رقم (٤):



الشكل رقم (٤)

٦. الآن نقوم باختبار الفيلم بالضغط على **Ctrl+Enter** .

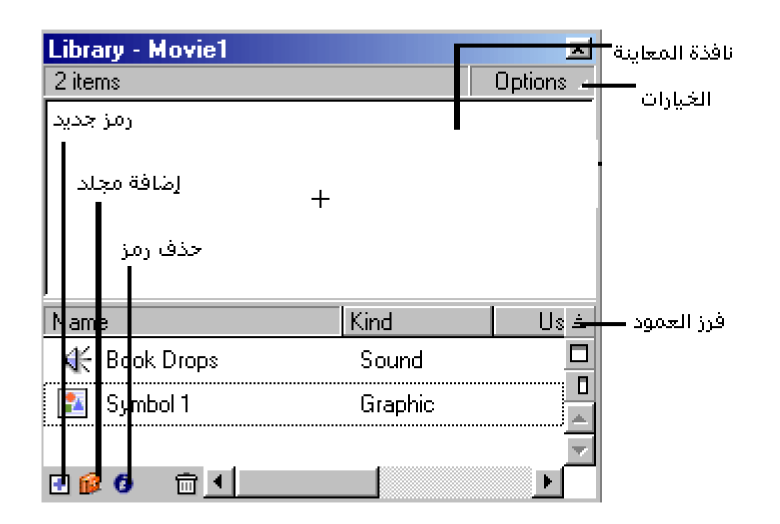
رابعاً- الطبقة المجلد:

فائدتها فقط هي تصنيف الطبقات ضمن مجلد ليسهل الوصول إليها .

التعامل مع المكتبات

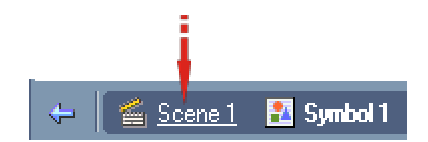
ستعلم في هذا الدرس بعض الأمور التي تسهل على المصمم العمل حيث تستخدم المكتبات لحفظ واستخدام جميع الوسائط، وكذلك استخدام القوالب المصممة مسبقاً وكذلك التعامل مع المكتبات المشتركة أعتبرها في نظري مهمة جداً، مع العلم أن استخدام الرموز يقلل من حجم الملف لأن فلاش لا يأخذ بعين الاعتبار عدد النسخ التي يتم توليدها من الرمز.

سنتعرف أولاً على نافذة المكتبة من الشكل رقم (١) التي تظهر من خلال الأمر **Library** من قائمة **Window** أو الضغط على **Ctrl+l**:



الشكل رقم (١)

يفضل دائماً إنشاء الرمز من خلال المكتبة من زر **Options** نختار الأمر **New Symbol** وذلك من أجل تحرير الرمز في نافذة مستقلة عن نافذة المشهد كما في الشكل رقم (٢) حيث نستفيد من وجود طبقات وشريط زمن مستقل لكل رمز مما يقلل من تعقيد أي فيلم، ثم نقوم بنسخها من المكتبة إلى نافذة الرسم:



الشكل رقم (٢)

و للانتقال إلى المشهد نضغط على **Scene** كما في الشكل السابق، ولا يمكن إضافة أي رسمة مستوردة أو منشأة ضمن فلاش إلى مكتبة المشروع إلا عن طريق الأمر **Convert to Symbol** من القائمة **Insert**،

ـ ولاستخدام عنصر موجود ووضعه ضمن الفيلم نختار أمر المكتبة **Library** من قائمة أمر النافذة **window** ، ثم نسحب العنصر المطلوب من المكتبة إلى نافذة الرسم.

وكافة الرموز المأخوذة من المكتبة لا يمكن تعديلها إلا عن طريقين:

١. كسر عناصر الرمز من خلال الأمر **Break Apart** من قائمة **Modify** أو الزر **Ctrl+B**.

٢. تعديل الرمز الرئيسي من نافذة المكتبة بالنقر المزدوج عليه مع ملاحظة أنه عند تغيير الرمز الرئيسي من المكتبة سوف يؤدي هذا الأمر إلى تعديل كافة النسخ المأخوذة لذلك الرمز.

ـ من أجل التحرير باستخدام برنامج خارجي:

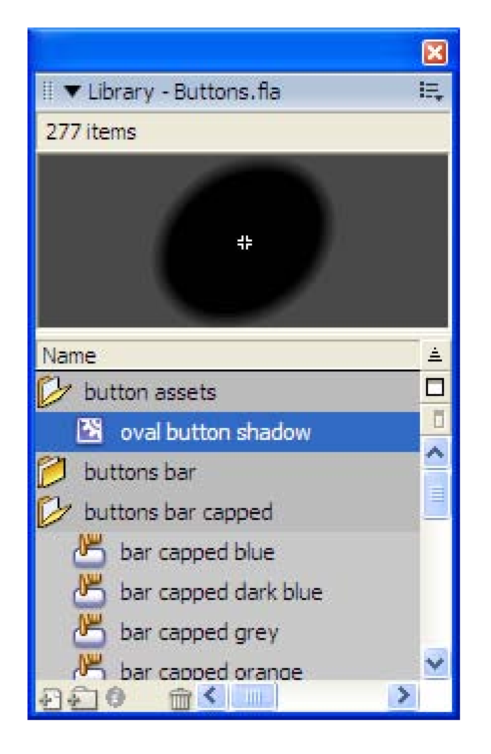
١. انقر على العنصر المطلوب لتحديده.
٢. اختر الأمر **edit with** من قائمة **options** يظهر مربع حوار انتقاء برنامج التحرير الخارجي **select external editor** .
٣. ابحث عن البرنامج المناسب ، ثم انقر **open** .
٤. قم بتحرير العنصر ثم احفظه، يقوم فلاش بتحديث العنصر ضمن المكتبة .

ـ تحديث العناصر المستوردة ضمن المكتبة :-

١. اختار الملف المستورد ضمن المكتبة .
٢. اختر أمر تحديث **update** من قائمة **options** .
٣. يظهر مربع التحديث **update media**، انقر على زر التحديث **update** .

المكتبة المشتركة في فلاش ٨:

يتضمن فلاش ٨ مجموعة كبيرة من الرسوم واللقطات والأزرار الجاهزة والفنية والأصوات ويمكن استعراض تلك الرسوم من خلال الأمر **Common Libraries** من قائمة **Window** كما في الشكل رقم (٣)



الشكل رقم (٣)

حيث يمكن استخدام المكاتب المشتركة من داخل أي مشروع من خلال سحب العناصر منها إلى مكتبة المشروع .

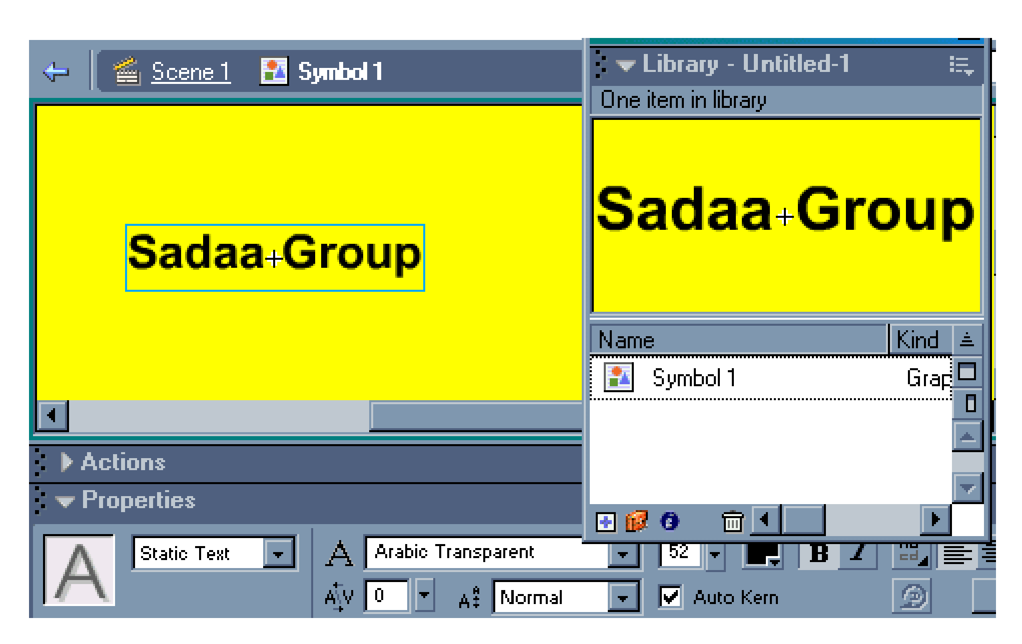
ـ إنشاء مكتبة خاصة أو جعل مكتبة المشروع مكتبة مشتركة لاستخدامها في مشاريع أخرى :-

و ذلك باتباع الخطوات التالية :-
١. قم بإنشاء أو استيراد العناصر التي تريد وضعها في المكتبة الخاصة .
٢. اختر أمر **save** من قائمة **file** يظهر مربع حوار حفظ الملفات باسم **save as** .
٣. ابحث عن مجلدات المكتبة **libraries** ، وهو المجلد الموجود ضمن المجلد الأساسي.
٤. اكتب اسماً للمكتبة ثم انقر **save**، حيث يتم حفظ المكتبة ضمن القائمة الفرعية **Common Libraries** من قائمة **Window** .

ـ لمعاينة اللقطة السينمائية المنشأة ضمن المكتبة والمنسوخة لنافذة الرسم من خلال الأمر **Test Movie** من قائمة **Control**.

مثال شامل فكرته تقوم على إنشاء نصوص بمقاييس مختلفة تتلاشى كلما كبرت ولكن سوف ننشئ هذا المثال مع الاستفادة من خصائص المكتبات وإنشاء القوالب وسوف يكون درساً ممتعاً لنبدأ:

١. نقوم بفتح نافذة المكتبة من خلال الضغط على Ctrl+l ومن قائمة Option التابعة لنافذة المكتبة نختار الأمر New Symbol تختار الرمز من نوع Graphic تفتح لنا نافذة مستقلة لتحرير الرمز نقوم بكتابة نص ما باستخدام أداة النص A وليكن النص Sadaa Group وذلك كما في الشكل رقم (١) حيث يمكن تغيير أعدادات النص من قائمة Text:

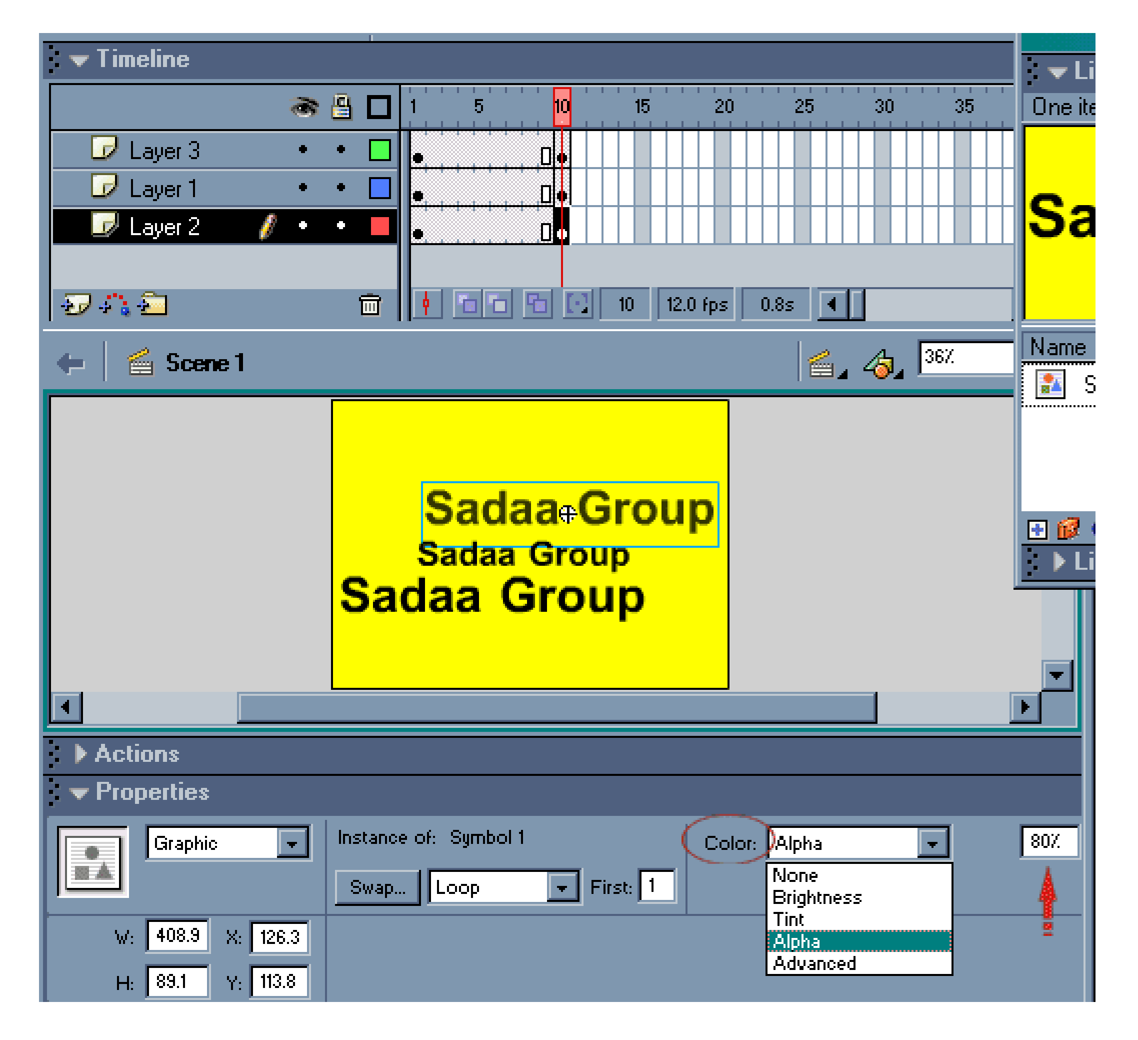


الشكل رقم (٢)

٢. نقوم بالعودة إلى نافذة تحرير المشهد بالضغط على رز Scene1 ثم نقوم بسحب الرمز من المكتبة إلى نافذة الرسم ونقوم بوضعه في وسط النافذة ثم نضيف إطار مفتاحي عند الإطار ١٠ بالضغط على مفتاح F6.

٣. نضيف الآن طبقتين ونجعل الطبقة التي أنشأنا عليها النص طبقة متوسطة بين الطبقتين الجديدتين وعلى كل طبقة نقوم بإضافة الرمز من المكتبة عند الإطار الأول ونحاول أن نجعل النصوص الثلاثة على مستوى واحد.

٤. نختار الآن الطبقة الدنيا عند الإطار ١٠ ونضيف إطار مفتاحي بالضغط على مفتاح F6 ثم نكبّر النص عند نفس الإطار من خلال أداة التحجيم نقوم بالسحب إلى الأعلى واليمين بمضاعفة حجم النص وكذلك الطبقة العليا ولكن إلى الأسفل واليسار كما في الشكل رقم (٣) ثم نقوم بإضافة تأثير Alpha لكل من الطبقة العليا والدنيا عند الإطار ١٠ من لوح Properties ونختار من قائمة Color الأمر Alpha ونضع القيمة تساوي الصفر كما في الشكل رقم (٣):



الشكل رقم (٤)

٥. الآن نضيف الحركة من نوع **Motion** من لوح **Properties** لكل من الطبقة العليا والدنيا عند الإطار ١ ثم ننفذ الحركة بالضغط على الزر **Ctrl+Enter** .

☜ *إن إحدى الفوائد من استخدام الكائنات النصية في فلاش هو إمكانية إعادة استخدام لمؤثر نصي تم بنائه مسبقاً دون الحاجة لإعادة إنشاؤه وتكون الخطوات هي:*

١. نفتح المؤثر المنشأ مسبقاً وليكن المثال الذي شرحناه سابقاً مع مراعاة أن يكون قد إنشأ بنفس ما قمنا بشرحه.

٢. ثم نقوم بفتح نافذة المكتبة بالضغط على **Ctrl+I** يظهر لنا الرمز **Symbol1** كما في الشكل السابق رقم (٢) نقوم بالنقر المضاعف بالماوس على ذلك الرمز لتحريره ضمن نافذته المستقلة كما في الشكل رقم (٢).

٣. باستخدام أداة النص A نقوم بالنقر على النص فيتيح لنا فلاش إمكانية إعادة كتابة النص نقوم بحذف النص وكتابة النص الجديد وليكن **Yasin Fares** ثم ننتقل إلى نافذة المشهد بالضغط على زر **Scene1**.

٤. نرى أن المشهد قد تحول إلى النص الجديد مع الحفاظ على نفس الحركة نقوم باختبار المشهد بالضغط على **Ctrl+Enter** .

ঔৣঔৣঔৣ

إدراج ملف الفلاش في صفحة ويب

إن إدراج فيلم الفلاش في صفحة ويب يعتبر من الأمور المهمة جداً ، حيث أن الفلاش أصبح يستخدم وبشكل كبير من أجل إضافة التأثيرات الحركية والجمالية في موقع الانترنت إضافة إلى الأكشن سكريبت وما تعطيه من إمكانيات تقنية

... ولذلك فإن تعلم هذا الأمر يعتبر من الأمور الأساسية وهذا الأمر بسيط جداً وهو ينقسم إلى قسمين إدراج ملف الفلاش عن طريق Front Page وإدراجه عن طريق لغة html ....

أدراج ملف الفلاش في Front page :-

١ - قم بعمل فيلم فلاش .

٢ - من القائمة الرئيسية للفلاش File أختر Export Movie احفظ الملف في القرص الصلب باسم sadaa على سبيل المثال بامتداد swf اضغط Save ثم OK ثم OK مرة أخرى .

*****لاحظ بأن البرنامج قد قام بتصدير الفيلم على هيئة Flash Player باسم sadaa.swf الملفات ذات الامتداد swf هي التي يمكن أن تدرج في صفحات الويب وتعرض في الإنترنت .***

٣ - الآن نقوم بفتح برنامج Front page ثم نفتح الصفحة اللي سنصدر العمل إليها ثم نختار من القائمة الرئيسة insert ← web component ← advansed ← Plug-in

أما بالنسبة لمستخدم Front page باللغة العربية يختار من القائمة الرئيسية إدراج ← مكون ويب ← عناصر تحكم متقدمة ← توصيل الشكل (١) .



الشكل (١)

٤ -الآن فتح لنا مربع حوار نقوم بتحديد الملف الذي سندرجه في Front page وذلك بالضغط على Browser ثم نختار الملف المطلوب ثم OK بعد ذلك قم بتحديد أبعاد الفيلم من طول وعرض .... الشكل (٢) .... هذا كل شيء في Front page ....



الشكل (٢)

إدراج ملف الفلاش عن طريق لغة html :-

طبعاً أولاً عليك القيام بالعمليتين السابقتين ١ ، ٢ حتى تقوم بحفظ الملف بامتداد swf .

<embed width="550" height="400" src="http://www.sadaagroup.com/sadaa.swf"

type="application/x-shockwave-flash">

انسخ الكود أعلاه وألصقه في المكان الذي تريد أن تدرج فيه الفيلم في صفحة الويب ، لا تنسى تغير وصلة الملف لتتلاءم مع عنوان موقعك الشخصي والمجلد الذي نقلت إليه ملف الفلاش إضافة إلى أبعاد فيلم الفلاش الخاص بك حيث أننا هنا وضعنا الأبعاد الافتراضية وهي ٤٠٠ X ٥٥٠ بكسل (لا تنسى الأبعاد بالبكسل) .

وهكذا تعلمت كيف تدرج فيلم فلاش في صفحة ويب وقد لاحظت أن هذا الأمر بغاية السهولة وخاصة إذا تم في

Front page

ßàßàßàßàßàßàßàßàßà

التعامل مع الأصوات في فلاش

إحدى المزايا التي يتمتع بها برنامج فلاش ويدعمها بقوة هي إمكانية إضافة الأصوات للأفلام حيث يتعامل فلاش مع نوعين من الأصوات (أصوات الأحداث، الأصوات المتدفقة) :-

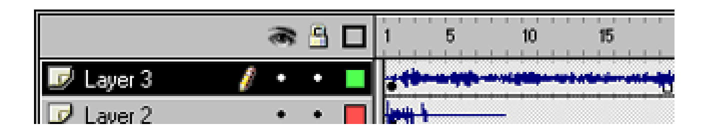
١ـ حيث تستخدم أصوات الأحداث للتركيز على أحداث معينة في الرسم ولا تعمل حتى يحمل الموقع والصوت بالكامل .

٢ـ أما الأصوات المتدفقة فتعمل مع عرض الرسم المتحرك وعند انتهاء عرض الرسم يتوقف الصوت فوراً حيث يحمل الصوت على دفقات وهذا النوع من الأصوات هو الأنسب لتصميم مواقع الإنترنت ولا انصح باستخدام الخيار Loop مع الأصوات المتدفقة الذي سوف نتعرف عليه لاحقاً ولإضافة صوت متدفق من خلال اللوح Properties نجعل خيار المزامنة Sync = Stream .

ـ ولاستيراد الصوت إلى فلاش من قائمة File نختار الأمر Import ونبحث على الملف المطلوب من نوع wav أو أي صيغة أخرى للصوت ثم نضغط Open، ويتم عرض الملف الصوتي ضمن المكتبة التي يمكن إظهارها من خلال الضغط على Ctrl+l ويضاف إلى اللوح Properties ضمن قسم Sounds لإضافته إلى الفيلم فيما بعد، ويفضل وضع كل صوت على طبقة منفصلة.

لانتقاء الصوت من المكتبة:

١. انتق الصوت عن طريق النقر على اسمه .
٢. استمع للصوت عن طريق النقر على زر التشغيل ply ضمن نافذة المكتبة .
٣. أضف الصوت إلى الفيلم عن طريق سحبه من مكتبة الأصوات إلى منطقة العمل بعد تحديد الإطار الذي تريد عنده إضافة الصوت نضيف إطار مفتاحي فارغ عن طريق الضغط على مفتاح F5.
٤. عندما تقوم بتعيين الصوت يظهر الشكل المتموج للصوت ضمن الإطار المفتاحي ثم يمتد عبر شريط الوقت ل ليغطي العدد الدقيق من الأطر التي سينطلق الصوت عبرها ، كما في شكل رقم (٢).



الشكل رقم (٢)

حيث يمكن تشغيل أصوات مختلفة في وقت واحد من خلال الإدراج ضمن إطارات متعددة كم في الشكل السابق .

التعامل مع الأزرار

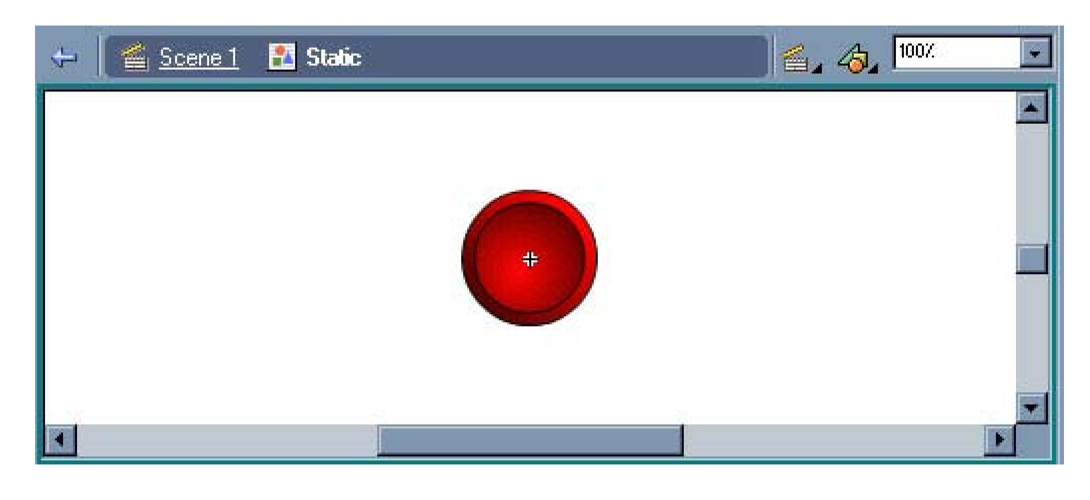
سنتعلم في هذا الدرس كيفية مبادئ تحويل الفيلم في فلاش من الشكل السكوني إلى الشكل التفاعلي حيث يستطيع المشاهد التحكم ببعض الأمور التي سوف تظهر في الفيلم وكل ذلك عن طريق الأزرار، ومن الأفضل دائماً أن تقوم بتصميم الجزء التفاعلي من بداية المشروع.

سنقوم أولاً بتعلم كيفية إنشاء زر متغير ثم كيفية ربط ذلك الزر بحدث معين من لوح Actions أو عن طريق استخدام أزرار جاهزة من المكتبة المشتركة Common Libraries.

إنشاء زر متغير:-

لإنشاء زر متغير نتبع الخطوات التالية:-

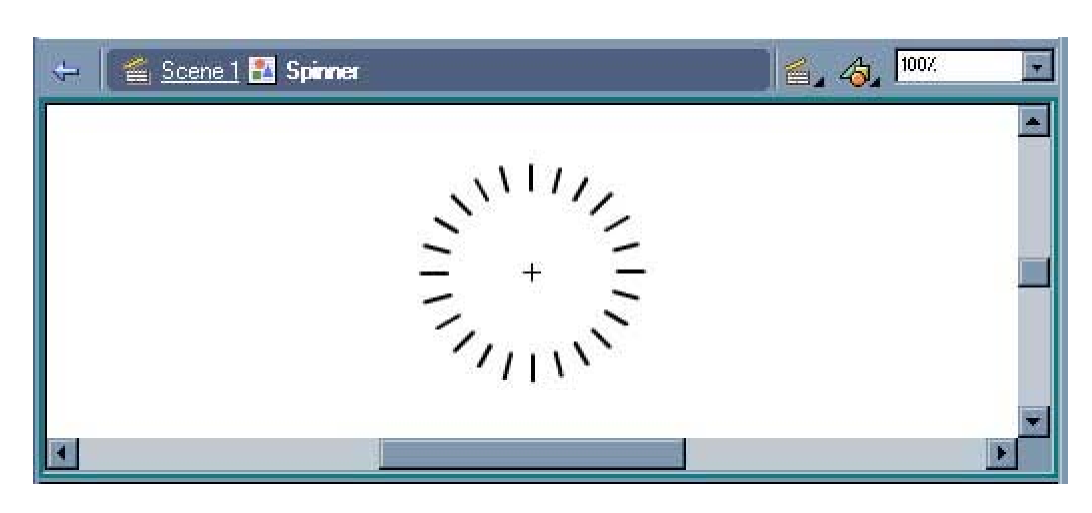
١. بعد فتح مشروع جديد وتجهيز أبعاده حسب المطلوب نقوم بإدراج رمز جديد من نوع Graphic من قائمة Insert نختار الأمر New Symbol ونسميه باسم Static ثم نقوم برسم زر أياً كان شكله مثلاً دائرة مجوفة كما في الشكل رقم (١):-



الشكل رقم (١)

ثم نقوم بالانتقال إلى المشهد الحالي بالضغط على زر Scene1 وبذلك نكون قد حفظنا الرمز ضمن مكتبة المشروع.

٢. الآن نقوم برسم التوهج للزر أثناء مرور الماوس من فوقه عن طريق إدارج رمز كما في السابق ولكن نسميه Spinner كما في الشكل رقم (٢):

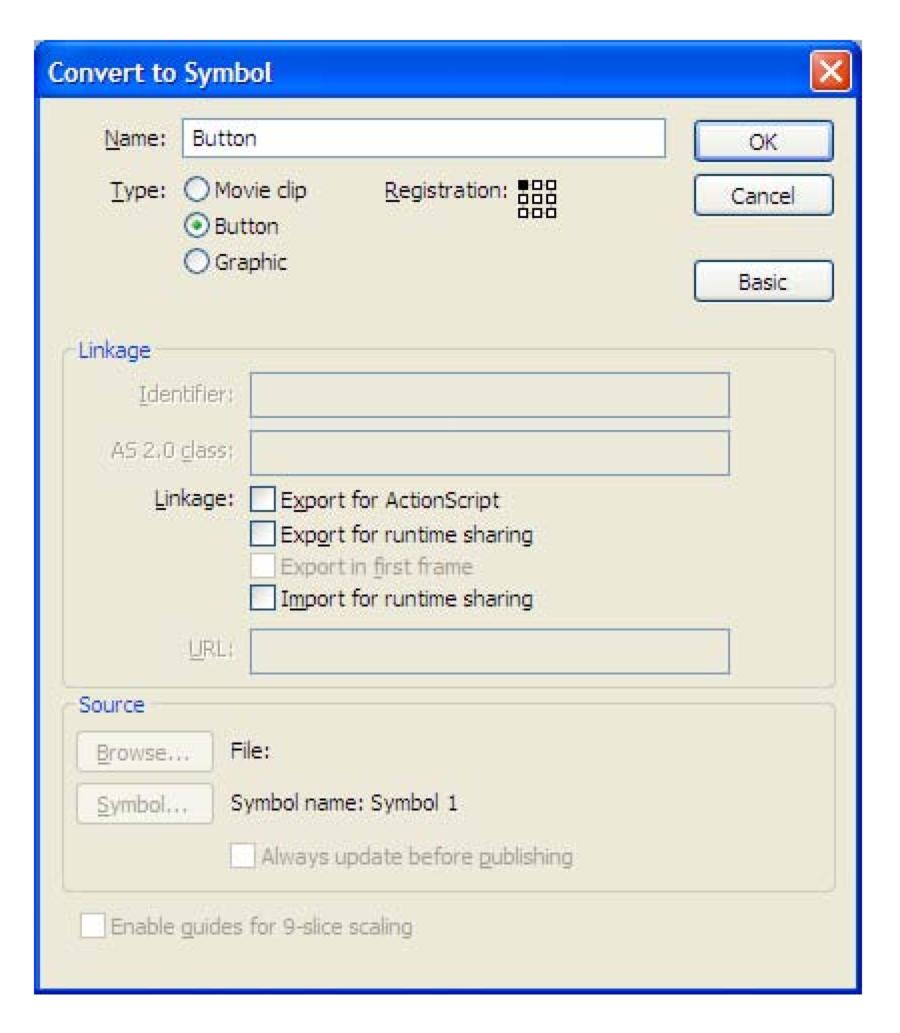


الشكل رقم (٢)

ثم نقوم بالانتقال إلى المشهد الحالي بالضغط على زر Scene1 وبذلك نكون قد حفظنا الرمز ضمن مكتبة المشروع.

٣. الآن نقوم بإنشاء لقطة سينمائية لحالة التوهج للزر عن إدراج رمز ولكن من نوع Movie Clip باسم Spinner Movie نفتح لنا النافذة الخاصة لإنشاء اللقطة السينمائية حيث نقوم بفتح نافذة المكتبة عن طريق الضغط على Ctrl+I ونختار رمز التوهج Spinner ونقوم بسحبه إلى نافذة اللقطة السينمائية عن طريق الماوس ونحاول توسيطة عند إشارة + ثم نقوم بإضافة إطار مفتاحي عند الإطار ١٠عن طريق الزر F6 ونضيف الحركة للرمز عن طريق النقر بالزر الأيمن للماوس عند الإطار ١ ونختار الأمر Create Motion Tween وعند الإطار ١٠ نحدد الرمز ونقوم بتدويره ربع دورة من خلال أداة التحجيم من خارج الإطار عندما يصبح شكل الماوس دائري نقوم بالتدوير وبذلك نكون قد أنهينا لقطة التوهج ننتقل إلى المشهد Scene1 .

٤. سوف نقوم الآن بإنشاء الزر المتغير وذلك عن طريق إدراج رمز من نوع Button من قائمة Insert نختار الأمر New Symbol ونسميه Button كما في الشكل رقم (٣):-



الشكل رقم (٣)

بذلك يتم فتح نافذة تحرير الزر ويكون في شريط الزمن أربع أطر نقوم بشرح مبسط عن كل إطار،

الحالات أربعة للأزرار:

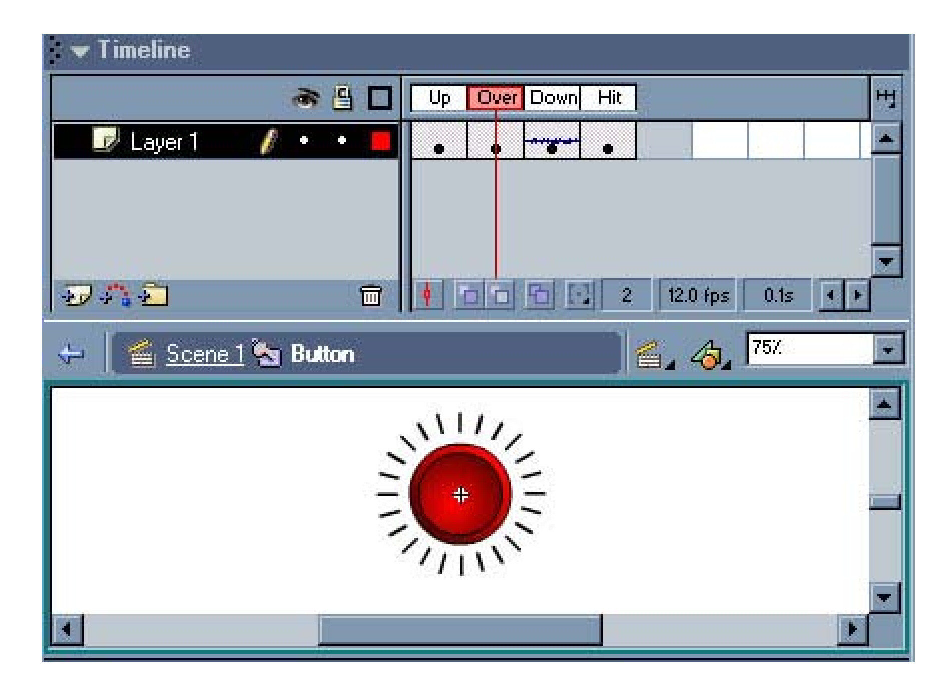
- حالة Up: غير منضغط ولا يكون مؤشر الماوس فوقه.

- حالة Over: تحدد الشكل الذي سيظهر عليه الزر عندما يكون مؤشر الماوس فوق المنطقة النشطة الخاصة بالزر.

- حالة Down: تحدد شكل الزر عندما يكون منضغطاً للأسفل.

- حالة Hit: تشير إلى المنطقة النشطة للزر المحدد ويجب أن يكون الشكل أكبر من باقي الأشكال ليتم تضمين باقي الأشكال فيها وعلى كل حال لن تكون تلك المنطقة مرئية في الفيلم النهائي، وإذا لم نحدد تلك المنطقة فإن فلاش يستخدم صورة الإطار Up لإطار Hit.

٥. نختار الإطارUp عن طريق الماوس ثم نفتح نافذة المكتبة ونسحب الرمز Static إلى نافذة تحرير الزر ونحاول توسيطة ثم نضيف أطر مفتاحية عند كل إطار عن طريق الزر F6، وعند الإطار Over نقوم بسحب اللقطة السينمائية Spinner Movie ونحاول توسيطها، وعند الإطار Down نقوم بقلب الزر لإعطاء حالة الإنضغاط للزر عن طريق أداة التحجيم نقوم بتدوير الدائرة نصف دورة ويمكن إضافة صوت من المكتبة المشتركة كما شرحنا في الدرس السابق عن طريق سحبه ووضعه ضمن إطار Down أيضاً، وبذلك تصبح نافذة تحرير الزر كما في الشكل رقم (٤):-



الشكل رقم (٤)

وبهذا نكون قد أنهينا الدرس نخرج الآن إلى المشهد **Scene1** ثم نقوم بمعاينة العمل بالضغط على **Ctrl+Enter** .

استيراد الصور النقطية والتعامل معها

هذا الدرس يفتقر إلى التطبيق العملي إلا أن المعلومات التي يتضمنها بالغة الأهمية فإن فلاش يسمح لك بإنشاء وتحريك الصور، أيضاً يسمح لك باستيراد ومعالجة الصور النقطية والصور المتجهة التي تم إنشاؤها في تطبيقات أخرى فمن المستحيل أن نقوم برسم كافة الرسوم التي نحتاجها وأحياناً نكون مضطرين لاستيراد بعض الصور اللازمة للفيلم، حيث أن معظم البرامج الرسومية تتعامل مع نوعين من الصور وهي (الصور المتجهة **Vector graphics** والصور النقطية **Bitmaps**) وسنوضح المصطلحين السابقين بشيء من الإيجاز وهما:

أولاً- الصور المتجهة: هي الصور التي تستخدم الخطوط والمنحيات التي تشكل المظهر الخارجي للصورة ويتم تحديد اللون من خلال لون خطوط الرسم، ويمكن تغيير خصائص الخطوط والمنحنيات التي تصف الشكل بدون أن يؤثر ذلك على جودة المنظر، حيث أن الصور المتجهة تملك خاصية استقلالية الدقة.

ثانياً- الصور النقطية: يتم توصيف الصورة النقطية باستخدام النقاط الملونة والتي تسمى بيكسل **Pixels** والتي يتم توزيعها على شبكة، وعند تحرير صورة نقطية يمكن تغيير البكسلات وليس الخطوط والمنحنيات، إن هذا النوع من الصور يملك خاصية عدم استقلال الدقة لآن المعلومات التي تصف الصورة مثبتة إلى شبكة بقياس محدد وبالتالي عند تحرير الصورة النقطية يمكن أن تتغير جودتها.

لاستيراد صورة نقطية إلى فلاش:-

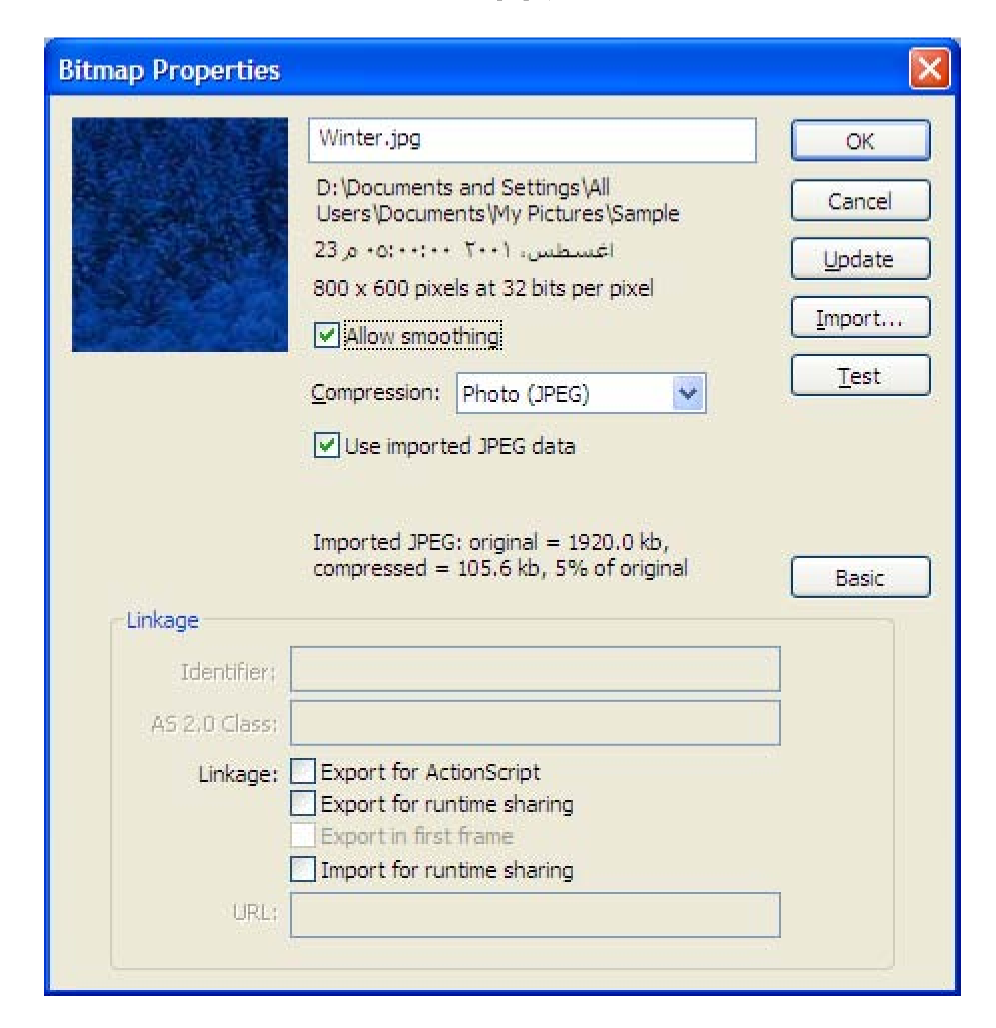
١. اختر أمر استيراد **Import** من قائمة **File** .
٢. ابحث عن الصورة الذي تريد استيراده .
٣. أنقر على زر الفتح **Open** لاستيراد الملف .

عند استيراد الصور تظهر في مكتبة المشروع.

- إن استيراد الصور النقطية يؤدي إلى زيادة حجم الملف التي يمكن تصغير حجمها بطريقتين:
*أولاً- خيار الضغط من المكتبة من قائمة **Options** نختار الأمر **Bitmap Properties**.*
لتحرير إعدادات تصدير الصورة النقطية:-

١. انتق الصورة المستوردة ضمن مكتبة المشروع.
٢. افتح قائمة خيارات لوح المكتبة Option Library ثم اختر أمر الخصائص Properties

انظر الشكل رقم (١).



الشكل رقم (١)

٣. خيار السماح بالنعومة Allow smoothing يكون فعالً بشكل افتراضي وهو يؤدي إلى تطبيق مفعول النعومة على الصورة .

٤. انقر على السهم الموجود في الطرف الأيمن من قائمة خيارات الضغط Compression ثم اختر Photo (JPEG أو Lossless(PNG/GIF .
٥. إذا انتقيت خيار الضغط الملائم للصور الفوتوغرافية فسيظهر قسم الجودة quality ضمن مربع الحوار ، أزل اشارة التحديد من مربع خيار استخدام الجودة الافتراضية للمستند use document default quality .
٦. أنقر على زر الاختبار test حيث يتم تحديث الصورة المصغرة ضمن نافذة المعاينة .

ثانياً- تحويلها إلى صورة متجهة نختار الأمر Trace Bitmap من قائمة Modify.

لتحويل الصور النقطية إلى عناصر رسومية:
١. حدد الصورة التي تريد تحويلها .
٢. اختر أمر تتبع الصورة Bitmap Trace من قائمة أمر المعالجة Modify لفتح مربع حوار تتبع الصورة كما في شكل (٢).

Trace Bitmap

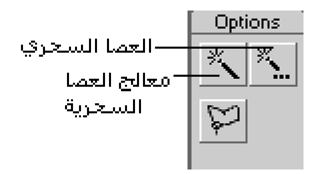
Color Threshold: 100 | OK
Minimum Area: 8 pixels | Cancel
Curve Fit: Normal
Corner Threshold: Normal | Help

الشكل رقم (٢)

٣. ضمن حقل نطاق الألوان Threshold Color ، اكتب قيمة تتراوح بين ١ و ٥٠٠ . وهذه القيمة هي التي تحدد مدى التقارب اللوني بين البكسلات المتجاورة في الصورة .
٤. ضمن حقل الحد الأدنى Area Minimum ، أكتب قيمة تتراوح بين ١ و ١٠٠٠ ، وهذه القيمة هي التي ت تحدد مقدار البكسلات المتجاورة .
٥. انتق خياراً من قائمة ملاءمة المنحنى Fit Curve . والخيارين المناسبين لمعظم الحالات Normal ، Smooth .
٦. انتق خياراً من قائمة نطاق الزاوية Threshold Corner ، هذا الإعداد هو الذي يحدد كيفية قيام فلاش ب برسم الزوايا . .
٧. أنقر موافق Ok لتحويل الصورة إلى عناصر رسومية .
يوضع الصورة النقطية الأصلية إلى اليسار ونسختين مستخلصتين إلى اليمين .

ـ وأفضل برنامج لعملة التحويل تلك هو برنامج (Free Hand)، مع تجنب تحويل الصور النقطية المعقدة لعدم وضوحها، ونلاحظ هنا أنه عند تحويل الصورة إلى رسوم متجهة أننا نستطيع تحرير الصورة وإزالة الأجزاء غير المرغوب بها كما سنرى في الفقرة التالية.

استخدام أمر الكسر والتفريق:-
الكسر والتفريق Break apart يعتبر طريقة أخرى من طرق تحويل الصورة النقطية إلى عناصر رسومية قابلة للتحرير .
لكسر وتفريق الصورة النقطية:
١. حدد الصورة النقطية التي تريد كسرها وتفريقها .
٢. اختر أمر الكسر والتفريق Apart Break من قائمة أمر المعالجة Modify ، بعد تفعيل الأمر ، يختفي صندوق الإحاطة الذي يحدد أطراف الصورة ، استخدم معالج العصا السحرية Wand Magic لانتقاء وتحديد المناطق اللونية المختلفة ، أو أداة القطارة لتحويل الصورة النقطية إلى لون داخلي .
٣. لانتقاء وتحديد المناطق اللونية المتماثلة بواسطة العصا السحرية قم بكسر وتفريق الصورة نقطية .
٤. انتق أداة الحبل lasso ، ثم انقر على magic wand كما في الشكل رقم (٣).



الشكل رقم (٣)

٥. انقر على زر magic wand properties لفتح مربع حوار إعدادات العصا السحرية .
٦. ضمن حقل النطاق Threshold ، اكتب قيمة تتراوح بين ٠ و ٢٠٠ وهذه القيمة تحدد مدى التقارب اللوني بين البكسلات المتجاورة .
٧. انتق أحد الخيارات من قائمة النعومة Smoothing وهذا الخيار هو الذي يحدد مدى نعومة حواف المنطقة المحددة .
٨. انقل الأداة نحو منطقة العمل ، لاحظ أن مؤشر الأداة يتحول من مؤشر الحبل إلى مؤشر العصا السحرية عند د

دخوله منطقة الصورة النقطية ، اضغط على مفتاح Shift ثم انقر على المزيد من المناطق اللونية لإضافتها إ ا الى التحديد، ثم ننقر نقراً مزدوجاً لإغلاق المضلع وبعد تفريغ الصورة يمكن إضافة حد خارجي للرسم لتغطية ا ا النتوءات الحاصلة باختيار لون الحد المناسب باستخدام أداة زجاجة الحبر

☞ أخيرا نقوم بتجميع الجزء المحرر باختيار الأمر Group من قائمة Modify.

ولتحويل الصورة النقطية إلى رمز عن طريق الأمر **Convert To Symbol** من قائمة **Insert** .

لتحويل الصورة النقطية إلى لون داخلي :-

١. حدد الصورة النقطية التي تريد استخدامها كعينة للون الداخلي .
٢. اختر أمر الكسر والتفريق Break apart من قائمة Modify .
٣. انتق أداة القطارة DROPPER ، ثم انقر على الصورة التي تم كسرها وتفريقها تتحول أداة القطارة إلى ا اللون الداخلي.

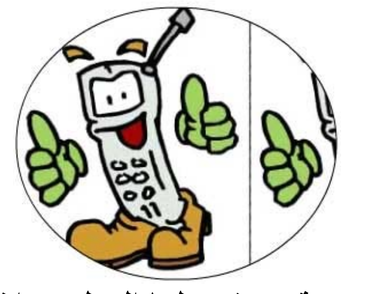
٤. بعد اخذ عينة من الصورة النقطية التي تم كسرها وتفريقها ، احذف الصورة ، وتذكر أن فلاش قد حفظ الصورة الأصلية ضمن مكتبة المستند .

لتطبيق مفعول اللون الداخلي المأخوذ من الصورة النقطية مع إمكانية تغير أبعاد الصورة، اتبع أحد الطرق التالية :

- بعد إنشاء اللون الداخلي من الصورة النقطية ، قم برسم عنصر ما على مسرح العمل ، وذلك إما أداة البيضاوي OVAL أو المستطيل rectangle ،يقوم فلاش بملء العنصر بلون العينة كما في الشكل رقم (٤).
- بعد إنشاء اللون الداخلي من الصورة النقطية انتق أداة الفرشاة ثم انقل الأداة نحو منطقة العمل لرسم بعض ا ا الضربات اللونية بالفرشاة وبلون النقش الذي تم استخلاصه من الصورة النقطية .
- لتعبئة العنصر الموجود باللون الداخلي المستخلص من الصورة النقطية ، أنتق أداة اللون الداخلي ثم انقر على ا العنصر.



الصورة الأساسية



الصورة بعد تحويلها إلى لون داخلي

الشكل رقم (٤)

وبذلك نكون قد تعلمنا كيفية التعامل مع الصور النقطية وتحريها والاستفادة منها في إعطاء الأفلام التي نقوم تصميمها حركة جمالية وفنية باستخدام الصور النقطية .

<u>لغة الاكشن اسكربت Action Script</u>

لغة الاكشن اسكربت Action Script هى مجموعة من الاجراءات و الاحداث المتسلسلة التى تخبر الكمبيوتر ماذا يفعل بالضبط .

نستفيد من لغة الاكشن اسكربت فى عمل الحركات Animation ، التنقل داخل الفيلم Navigation ، التعامل مع مدخلات المستخدم لملف الفلاش User Input ، التعامل مع البيانات Get Data ، عمل العمليات الحسابية Calculations ، العمل مع ملفات الصوت Play Sounds، و عمليات اخرى كثيرة .

بداية نتسأل فى اى عناصر داخل الفلاش يتم كتابة اكواد لغة الاكشن اسكربت ؟

١- الخط الزمنى Scripts in the Timeline .

٢- الازرار Scripts Attached to Buttons.

٣- الموفى كليب.Scripts Attached to Movie Clips.

<u>تاريخ لغة الاكشن اسكربت:</u>

ActionScript ١.0 من الاصدار الرابع من برنامج فلاش و حتى الاصدار السادس

Action Script 2.0 من الاصدار السابع حتى الاصدار الثامن .

Action Script 3.0 من الاصدار التاسع .

<u>ملاحظة :</u> **قد نرمز للغة الاكشن اسكربت بالرمز AS 3.0**

الفصل الاول

اساسيات الاكشن اسكربت Action Script 2

يمكننا كتابة كود الاكشن اسكربت من خلال محرر الكود الخاص بهذة اللغة و هو مدمج داخل البرنامج و يمكننا فتحة من خلال النافذة actions و لفتح اللوحة Actions انقر على القائمة windows اختر actions او اضغط f9 من لوحة المفاتيح .

و يتم تنفيذ الكود من خلال اضافتة الى الكادرات Frames او القطع الحركية MovieClip او الازرار التى يتم انشائها داخل العمل .

لكن لاحظ ان يتم تنفيذ الامر عند وقوع حدث معين على هذا الزر مثال على هذة الاحداث الخاصة بالازرار Buttons :-

الحدث Event	شــــرح الــحــدث
Press	سوف يتم تنفيذ الامر المرتبط بهذا الحدث عندما يقوم المستخدم بالضغط على مفتاح الفأرة الايسر و قبل عودتة للوضع الطبيعى .
Release	سوف يتم تنفيذ الامر المرتبط بهذا الحدث عندما يقوم المستخدم بالضغط على مفتاح الفأرة الايسر ثم ترك مفتاح الفأرة مع وجود المؤشر فوق المفتاح
releaseOutside	سوف يتم تنفيذ الامر المرتبط بهذا الحدث عندما يقوم المستخدم بالضغط على مفتاح الفأرة الايسر ثم ترك مفتاح الفأرة ابتعاد المؤشر عن المفتاح
keyPress	سوف يتم تنفيذ الامر المرتبط بهذا الحدث عندما يقوم المستخدم بالضغط على مفتاح فى لوحة المفاتيح .
rollOver	سوف يتم تنفيذ الامر المرتبط بهذا الحدث عندما يقوم المستخدم بتمرير المؤشر فوق الزر .
rollOut	سوف يتم تنفيذ الامر المرتبط بهذا الحدث عندما يقوم المستخدم بتمرير المؤشر فوق الزر و الابتعاد بالمؤشر عن الزر.
dragOver	سوف يتم تنفيذ الامر المرتبط بهذا الحدث عندما يقوم المستخدم بالضغط على الزر و سحب الزر مع الضغط فوقة .
dragOut	سوف يتم تنفيذ الامر المرتبط بهذا الحدث عندما يقوم المستخدم بالضغط على الزر اثناء وجود المؤشر فوق الزرثم تحريك المؤشر بعيدا عن الزر

الاحداث الخاصة بال movieClip

شـــرح الـحــدث	Event الحدث
حدث تحميل القطع الحركية	Load
حدث خروج القطع الحركية من مسرح العمل	Unload
عند بداية تحميل الفريم الاول للموفى كليب و يستمر حتى ينتهى الموفى كليب .	enterFrame
عند الضغط على زر الفأرة الايسر	mouseDown
عند تحرير زر الفأرة الايسر	mouseUp
عندما يتحرك مؤشر الفأرة	mouseMove
عند الضغط على مفتاح من لوحة المفاتيح	keyDown
عند تحرير الضغط عن مفتاح من لوحة المفاتيح	keyUp
	Data

- اعتقد انة حان الوقت لكى نجرب اول اسكربت لنا مع هذة اللغة:
- قم بعمل فيلم جديد ثم بعد ذلك قم بعمل ثلاث كادرات **key frames** و ذلك بالضغط على الفريم الاول فى الشريط الزمنى ثم الضغط على **f7** من لوحة المفاتيح مرتين و باستخدام مهاراتك العادية التى اكتسبتها من العمل على برنامج فلاش قم بعمل ثلاث اشكال مختلفة كل شكل فى فريم خاص بة من الثلاث فريمات ، الان قم بعمل تشغيل تجريبى للفيلم من خلال الضغط على مفتاحى **ctrl+enter** من لوحة المفاتيح ، سوف تشاهد الفيلم يعيد تكرار نفسة كلما انتهى.
- ارجع للملف الاصلى و قف بالمؤشر عند الفريم الثانى و اضغط **f9** فيظهر لك نافذة محرر كود الاكشن اسكربت و اكتب **stop();** . و اغلق النافذة و قم بتجربة العمل

سترى ان الفيلم لم يعرض منة الا الفريم الاول و الثانى ثم بعد ذلك توقف و لم يكرر نفسة من جديد و من هنا نكون رأينا اول فائدة من هذة اللغة و هى التحكم فى الفيلم .

كـيـف يـفـكـر الـكـمـبـيـوتـر؟؟

فى الحقيقة الكمبيوتر لا يفكر بل يأخذ التعليمات و ينفذها بحذافيرها اى لا يضيف او يفكر هل التعليمات التى قمت بأعطائها له منطقية ام لا .

و لغة الاكشن اسكربت هى مجموعة من التعليمات تكتب فى اسكربت داخل الفلاش لكى تخبر الكمبيوتر ماذا يفعل بالضبط .

و لكى تسطيع تعلم البرمجة يجب ان تفكر كالكمبيوتر كل امر تكتبة فكر بة كأنك انت الكمبيوتر او بمعنى ادق كأنك برنامج فلاش و تصور تسلسل هذة الاوامر و التعليمات و فكر كيف تنفذها و ما هو ناتجها .

المبرمج الحقيقى ليس فقط من يستطيع كتابة كود برنامجة من البداية للنهاية بل ايضا من يستطيع فهم اكواد الاخرين و تعديلها للأفضل يجب ان تتعلم من الاخرين و تحلل الافكار لكى تكون قادرا على طرح افكارا جديدا و لكى لا ترهق نفسك فى التفكير فى حلول لمشاكل سبقك اخرين لحلها .

مثال سوف نقوم بتحليل كود قمت بكتابتة و سوف تستفيد من هذا التحليل كثيرا عندما تنتهى من قراء ة الكتاب ككل

```
on (press) {
   var myVariable:Number = 7;
   var myOtherVariable:String = "Macromedia";
   for (var i:Number=0; i<10; i++) {
      trace(i);
      if (myVariable + 3 == 5) {
         trace(myOtherVariable);
      }
   }
}
```

السطر الاول تحليلة اننى قمت بعمل زر فى الفلاش و قمت بكتابة كود لا يتم تنفيذة الا عند الضغط على الزر والامر المسئول عن ذلك هو **on (press)** و لا حظ انه لا يمكن استخدام الامر السابق الا مع الازرار .

اما كل ما هو مكتوب بين العلامتين **{ }** هو كود يتم تنفيذة عند الضغط

السطر الثانى قمت بتعريف متغير رقمى اسمة **myVariable** و اسندت لة القيمة ٧ .
السطر الثالث قمت بتعريف متغير نصى اسمة **myOtherVariable** و اسندت لة القيمة **Macromedia** .
السطر الرابع استخدمت التكرار من خلال عبارة **for** .
السطر الخامس استخدمت الدالة **trace** لرؤية المخرجات .
السطر السادس استخدمت التحكم فى الكود من خلال **if** .
فى النهاية قم بتجربة الكود و سترى النتيجة تظهر فى شاشة المخرجات التى نستخدمها لرؤية تأثير الكود على الفيلم و سوف نناقش الاسكريبت السابق فى نهاية الكتاب و سترى بنفسك انك بعد الدروس التالية ستكون قادرا على تحليل الكود و شرحة و تطويرة بنفسك.

الـفـصـل الـثـانــى

البيانات و أنواعها Data and Data Types

هذا الفصل سوف نتحدث فيه عن مبادئ لغة الاكشن اسكربت ٢ و سوف نتعلم فى هذا الفصل كيف نتعامل مع البيانات و ما هو نوع البيانات التى يمكننا التعامل معها من خلال هذة اللغة.
البيانات عموماً هى مجموعة من الحروف و الارقام و المعلومات و يمكنك توصيف او تحديد نوع البيانات من خلال المتغيرات و هذا يدفعنا للتسائل *ما هو المتغير ؟!*

المتغيّرات Variables:

كما في أيّ لغة برمجة، تقوم المتغيّرات بتخزين القيمِ أثناء تنفيذ البرنامج.. وطبعًا سُمّيت متغيّراتٍ، لأنّك تستطيع تغيير قيمها في أيّ لحظةٍ أثناء تنفيذ البرنامج.
وللمتغيّر اسمٌ وقيمة.. *فمثلا:* المتغيّر "اسم المستخدم" **UserName** يمكن أن تُوضع به القيمة "مايكل".. والمتغيّر "الخصم" **Discount** يمكن أن تُوضع به القيمة ٠,٤٠.
تلاحُظ هنا أنّ القيمتين "مايكل" و ٠,٤٠ مختلفتان، فالأولى نص **String** لهذا تمّ وضعها بينَ علامتَيْ تنصيص، بينما الثانية قيمة رقميّة **Numeric Value**.
المتغيّرات في الاكشن اسكربت ٢ ليست مجرّد أسماءٍ أو مخازن للقيم.. إنّها كذلك كِيانات ذكيّة لتخزين وإجراء العمليات على القيم.. باختصار: إنّها كائنات **Objects**، لها وسائلها وخصائصها الخاصة بها.

أنواع البيانات Types of data

String *النّصّيّة:* تم شرحها فى المثال السابق.

Number *الرقميّة:-*تم شرحها فى المثال السابق .

Boolean *المنطقية :*
البيانات المنطقيّة تخزّن واحدةً فقط من القيمتين: "صواب" **True** و"خطأ" **False**، وهي في الأساس أعداد صحيحة، فالقيمة "صواب" تعادل ١، والقيمة "خطأ" تعادل صفرا.. وفي الواقع، أي قيمة غير صفرية، تعتبر **True**.

Object *الكائنات:-*
MovieClip مقطع فيلم لة خصائص و أحداث
Null قيمة فارغة
Undefined قيمة غير معرفة

تعريف المتغيرات Declaring Variables

لماذا نعرّف المتغيّر؟؟؟؟؟؟

في معظم لغاتِ البرمجة، يجبُ تعريفُ المتغيّراتِ أولا قبلَ استخدامها.. إنّ هذا يجعل الأمرَ أيسرَ بالنسبة لمترجم الكود Compiler، ففي كلّ مرّة يصادف المترجم متغيّرا، عليه أن يُنشئه في الذاكرة، ونتيجة لاعتباراتٍ في تنظيم الذاكرة، فإن مثل هذه العملية تستهلك بعض الوقت، ممّا يُبطئ البرنامج.. ولكن لو كان المترجم يعرف كل متغيرات البرنامج وأنواعها سلفا قبل أن يبدأ ترجمة البرنامج، ففي هذه الحالة سيتحسّن الأداء لأقصى درجة.

لقد كانت من أشهر سمات اكشن اسكربت (١)، عدم إرغامه للمبرمج على تعريف كلّ المتغيرات.. لقد صارت هذه السمة منتقدةً الآن بشدّةٍ، ليس فقط للأسباب المتعلّقة بسرعة الترجمة وكفاءة الأداء، ولكن أيضا لأن تعريف المتغيّر يُمكّن المترجم من اصطياد أخطاء كثيرةٍ، سواء في وقت التصميم Design Time أو وقت الترجمة Compile Time، بدلا من أن تُفاجئكَ في وقت التشغيل Runtime.

الشروط الواجب توافرها عند تعريف المتغيرات:

- ألا يكون كلمة من كلمات اللغة الأساسيّة (تلك التي تراها باللون الأزرق عند فتح لوحة actions).
و منها :

- **break**
- **case**
- **class**
- **continue**
- **default**
- **delete**
- **dynamic**
- **else**
- **extends**
- **for**
- **function**
- **get**
- **if**
- **implements**
- **import**
- **in**
- **instanceof**
- **interface**
- **new**
- **private**
- **public**
- **return**
- **set**
- **static**
- **switch**
- **this**
- **typeof**
- **var**
- **void**
- **while**
- **with**

- ألا يزيد عن ٢٥٥ حرفًا، وهو رقم كبير بالفعل بما يكفي.
- أن يتكون من كلمة واحدة لا تتخللها المسافات.. ويمكن استخدام الشرطة المنخفضة "_" للفصل بين مقاطع الكلمة بدلا من المسافات.
- لا يبدأ بأرقام، وإن كان من الممكن أن تتوسطه أرقام، أو ينتهيَ بها.
- لا يحتوي على أيٍ من: علامات التنصيص أو الأقواس أو النقطة "."، ولا علامات العمليات الحسابية أو علامات المقارنة الحسابية أو المنطقية، فكل هذه العلامات محجوزة لوظائف أخرى.
- غير مسموح بتكرار اسم المتغيّر داخل نفس النطاق، فلا يمكن تعريف متغيّرين متماثلين في الاسم داخل نفس الإجراء، وإن كان من الممكن تكرار نفس اسم المتغيّر لكن في إجراءات مختلفة.

-والمتغيرات في لغة الاكشن اسكريبت (٢) لا تتجاهل حالة الأحرف **Case-insensitive**، فالأسماء **myAge** و **myage** و **MYAGE**، ليست كلّها متكافئة معنى هذا أنّك تستطيع استخدام هذه الكلمات لتعريف ثلاثة متغيرات مختلفة، فكلّها لا تُعتبر اسما واحدا.

طريقة تعريف المتغير:-

Var اسم المتغير: نوع المتغير

كلمة **Var** هى المسئولة عن حجز مكان فى الذاكرة لمتغير ثم نقوم بأعطاء المتغير اسم و نوع بعد ذلك مثال:

```
var sProduct:String;
```

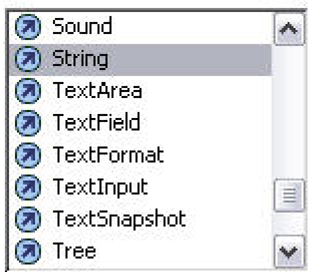
بعد تعريف المتغير يمكنك اسناد قيمة ما لهذا المتغير شرط ان تكون من النوع الذى عرفتة للمتغير

```
sProduct = "car";
```

و يمكن اختصار الجملتين فى جملة واحدة هكذا

```
var sProduct:String = "car";
```

لاحظ انه عندما تكتب **var** ثم اسم المتغير ثم : فيظهر لك نافذة بها أنواع البيانات التى قد تحتاجها



اذا كنت تريد عرض قيمة المتغير **sProduct** فى وضع الاختبار يمكنك استخدام الدالة **trace()** فسوف تقوم بارسال القيمة الى شاشة المخرجات **Output panel** فيصبح الكود كالتالى :

```
var sProduct:String = "car";

trace(sProduct);
```

الناتج

```
// car
```

لو قمت بتغيير قيمة المتغير **sProduct** الى ١٢٠ مثلا :-

```
sProduct = 120
```

سوف ترى الخطأ التالى :-

Type mismatch in assignment statement: found Number where String is required.

هذا الخطأ يخبرك انك قمت باسناد بيانات خطأ لمتغير معين . لاحظ ان علامة التنصيص (" ")

the quotation marks ليست ضرورية فى حالة المتغيرات الرقمية

```
var numWrinkles:Number = 55;
```

و اذا كنت تريد تغير قيمة المتغير بعد ذلك يمكنك كتابة الكود التالى :-

```
numWrinkles = 60;
```

اذا كانت القيمة رقمية او منطقية (صح او خطأ) لا تستخدم .(quotation marks) مثال :-

```
var age:Number = 38;

var married:Boolean = true;

var hasChildren:Boolean = false;
```

ما هى المصفوفة array:-

البرمجة الحقيقيّة تبدأ من هذه النقطة، فلقد صُنع الكمبيوتر أساسا، ليقوم بالعمليات الرتيبة المتكرّرة لآلاف أو ملايين المرّات، بسرعةٍ وبدون ملل.
افترض مثلا أنّك تريد حساب متوسط العمر لخمسين طالبا.. أوّل ما ستفكّر فيه، هو أن تعرّف خمسين متغيّرا وتجمعها معا وتقسم الناتج على ٥٠.. إنّ مثل هذه الطريقة ستستهلك منك شهرا على الأقلّ لكتابتها، وهي كفيلة بجعلك تكره البرمجة اساسا!
مع أنّك تستطيع أن تكتب هذا البرنامج في خمس سطور لا غير.. تخيّل!
والفكرة كلّها تعتمد على تخزين أعمار الطلبة في "تركيب ما"، يمكن للكمبيوتر أن يتعامل معه بطريقة آليّة رتيبة متكرّرة، لينفّذ عليه العمليّات التي نريدها.
هذا التركيب هو المصفوفة Array، التي تتكوّن من مجموعة من الخانات، كل خانة منها تحتفظ بقيمة معيّنة، بحيث يمكن الوصول لهذه القيمة عن طريق رقم خانتها Index.
مثال :

```
var childrenArr:Array = new Array("Pylon", "Smithers", "Gil");
```

لاحظ أن يمكنك كتابة الكود السابق بطريقة مختصرة كما يلى :-

```
var childrenArr:Array = ["Pylon", "Smithers", "Gil"];
```

و هناك طريقة اخرى لتعريف مصفوفة تحتوى اسماء الشهور :

```
var myArr:Array = new Array();

myArr[0] = "January";
myArr[1] = "February";
myArr[2] = "March";
myArr[3] = "April";
```

بنفس الطريقة يمكننا تعريف كائن جديد (object) مثال :

```
var myObj:Object = new Object();
myObj.firstName = "Steve";
myObj.age = 50;
myObj.childrenArr = new Array("Mike", "Robbie", "Chip");
// طريقة اخرى
```

```
var myObj:Object = {firstName:"Steve", age:50, childrenArr:["Mike",
  "Robbie", "Chip"]};
```

ملحوظة : **يمكنك استخدام العلامة // فى نافذة محرر الكود لكى نوضح لبرنامج محرر الكود فى فلاش ان الكلمات التى تلى هذة العلامة ليست اكواد و لكنها تعليقات لتذكرة المبرمج بسبب كتابة للكود اى هوامش لا يتم تنفيذها .**

كود أكشن اسكربت ٣,٠ لعمل مصفوفة Array

```
var array:Array = new Array();

var array:Array = new Array(elements);

var array:Array = new Array(element0,...elementN);
var letters:Array = ["a", "b", "c"];
items[4] = "apples";
trace(items[4]);
```

لاحظ ان اكشن اسكربت لا تهتم بنوع القيم الموجودة داخل المصفوفة
مثال هذة المصفوفة تحتوى على انواع مختلفة من المتغيرات

```
 string, integer, a Boolean, and an object:
var data:Array = ["a", 2, true, new Object()];
```

اضافة عناصر لبداية المصفوفة و نهايتها

```
var array:Array = new Array();
array.push("val 1", "val 2");
```

و يمكن اضافة عنصر اخر من خلال معرفة طول المصفوفة

```
array[array.length] = "val 3";
```

مثال عمل مصفوفة مكونة من اربعة عناصر و اضافة عنصر جديد z فى بداية المصفوفة لذلك القيمة a سوف تتحرك من بداية المصفوفة اى من الترتيب (٠) داخل فهرس المصفوفة الى الترتيب (١) و هكذا

```
var letters:Array = new Array( );
letters.push("a", "b", "c", "d");
letters.unshift("z");
```

و لعرض عناصر المصفوفة نقوم بعمل جملة تكرار بالكود التالى :

```
for (var i:int = 0; i < letters.length; i++) {
  trace(letters[i]);
}
```

عرض محتويات مصفوفة من خلال جملة for

```
var letters:Array = ["a", "b", "c"];
for (var i:int = 0; i < letters.length; i++) {
    trace("Element " + i + ": " + letters[i]);
}
```

عرض محتويات المصفوفة بالعكس :

```
var letters:Array = ["a", "b", "c"];
for (var i:int = letters.length - 1; i >= 0; i--){
    trace("Element " + i + ": " + letters[i]);
}
```

يمكنك استخدام طول المصفوفة باسناد القيمة الى متغير مثال :

```
var length:int = sprites.length;
for (var i:int = 0; i < length; i++){
   sprites[i].x++;
```

```
}
```

البحث عن قيمة عنصر معين داخل المصفوفة :

```
// قم بعمل مصفوفة مكونة من ٨ عناصر
var letters:Array = ["a", "b", "c", "d", "a", "b", "c", "d"];

// حدد القيمة المراد البحث عنها و ضعها فى متغير نصى
var match:String = "b";

// قم بعمل جملة for لعرض محتويات المصفوفة
for (var i:int = 0; i < letters.length; i++) {

    // تأكد من ان القيمة المراد البحث عنها متتطابقة ام لا
    if (letters[i] == match) {

        اذا وجدت العنصر المراد البحث عنة اظهر رسالة
        trace("Element with index " + i +
             " found to match " + match);

// استخدم كلمة break للخروج من جملة التكرار
        break;
    }
}
```

قم بتجربة المثال التالى :

```
var letters:Array = ["a", "b", "c", "d", "a", "b", "c", "d"];

var match:String = "b";

// Loop backward through the array. In this example,
// the "b" is at index 5.
for (var i:int = letters.length - 1; i >= 0; i--) {
    if (letters[i] == match) {
        trace("Element with index " + i +
             " found to match " + match);
        break;
    }
}
```

يمكنك تبسيط عملية البحث من خلال استخدام الكلاس

//The class is in the *ascb.util* package

```
import ascb.util.ArrayUtilities;

var letters:Array = ["a", "b", "c", "d"];

trace(ArrayUtilities.findMatchIndex(letters, "b"));
// Displays: 1
```

```
trace(ArrayUtilities.findMatchIndex(letters, "r"));
// Displays: -1
```

ملحوظة كلاس المصفوفة بها ٣ وظائف

findMatchIndex( ), findLastMatchIndex( ), and findMatchIndices( ).

مثال :

```
public static function findMatchIndex(array:Array, element:Object):int {
   // استخدم متغير لتحديد بداية الفهرس
   //  اختر ان تبدأ من الصفر
   var startingIndex:int = 0;

   // By default don't allow a partial match.
   var partialMatch:Boolean = false;

   // If the third parameter is a number,
   // assign it to nStartingIndex.
   // Otherwise, if the fourth parameter is a number,
   // assign it to nStartingIndex instead.
   if(typeof arguments[2] == "number") {
      startingIndex = arguments[2];
   }
   else if(typeof arguments[3] == "number") {
      startingIndex = arguments[3];
   }

   // If the third parameter is a Boolean value,
   // assign it to partialMatch.
   if(typeof arguments[2] == "boolean") {
      partialMatch = arguments[2];
   }

   // Assume no match is found.
   var match:Boolean = false;

   // Loop through each of the elements of the array
   // starting at the specified starting index.
   for(var i:int = startingIndex;
         i < array.length; i++) {

      // Check to see if the element either matches
      // or partially matches.
      if(partialMatch) {
         match = (array[i].indexOf(element) != -1);
      }
      else {
         match = (array[i] == element);
```

```
    }

    // If the element matches, return the index.
    if(match) {
      return i;
    }
  }

  // The following return statement is only reached
  // if no match was found. In that case, return -1.
  return -1;
}
```

ازالة عنصر من المصفوفة :

لازالة عنصر من منتصف المصفوفة استخدم الوظيفة splice( )

لازالة عنصر من نهاية المصفوفة استخدم الوظيفة pop( )

لازالة عنصر من بداية المصفوفة shift( )

مثال :

```
var letters:Array = ["a", "b", "c", "d"];

// Remove the first element and display its value.
trace(letters.shift( ));

// Remove the last element and display its value.
trace(letters.pop( ));

// Display the remaining elements.
// The array has two elements left: "b" and "c".
for (var i = 0; i < letters.length; i++) {
  trace(letters[i]);
}
```

عندما تُزيلُ عنصر من المصفوفةَ ، تَحتاجُ لتَغيير قيمةِ متغيّرِ الفهرس وفقاً لذلك. يُصوّرُ المثالُ التاليُ ما يَحْدثُ إذا لم تُجدّدْ قيمة متغيّرِ الفهرسَ:

```
var numbers:Array = new Array(4, 10);
numbers[4] = 1;
trace(numbers);  // Displays: 4,10,undefined,undefined,1
for(var i:int = 0; i < numbers.length; i++) {
  if(numbers[i] == undefined) {
    numbers.splice(i, 1);
  }
}
trace(numbers);  // Displays: 4,10,undefined,1
```

لكن الطريقة الصحيحة كما يلى

```
var numbers:Array = new Array(4, 10);
numbers[4] = 1;
trace(numbers);  // Displays: 4,10,undefined,undefined,1
for(var i:int = 0; i < numbers.length; i++) {
```

```
      if(numbers[i] == undefined) {
        numbers.splice(i, 1);
        i--;
      }
    }
    trace(numbers);  // Displays: 4,10,1
```

يمكنك ايضا حف عنصر فى مصفوفة و اضافة اخر فى نفس الوقت كما يلى :

```
var letters:Array = ["a", "b", "c", "d"];

// Remove two elements and insert three more
// into letters starting at index 1.
letters.splice(1, 2, "r", "s", "t");

// myArray now contains five elements:
// "a", "r", "s", "t", and "d".
for (var i:int = 0; i < letters.length; i++) {
   trace(letters[i]);
}
```

تحويل متغير نصى الى مصفوفة :

```
var list:String = "Peter Piper picked a peck of pickled peppers";
// استخدم المسافة للفصل بين الكلمات فى السطر السابق و وضع كل كلمة كعنصر فى المصفوفة
var words:Array = list.split(" ");
```

للتحويل من مصفوفة الى متغير نصى :

```
var letters:Array = ["a", "b", "c"];
trace(letters.join("|"));   // Displays: a|b|c
```

مثال اخر :

```
var letters:Array = ["a", "b", "c"];
trace(letters.join());   // Displays: a,b,c
```

ترتيب المصفوفة :

```
var words:Array = ["tricycle", "relative", "aardvark", "jargon"];
words.sort( );
trace(words); // Displays: aardvark,jargon,relative,tricycle
var scores:Array = [10, 2, 14, 5, 8, 20, 19, 6];
scores.sort( );
trace(scores);   // Displays: 10,14,19,2,20,5,6,8
```

لعمل ترتيب عددى

```
var scores:Array = [10, 2, 14, 5, 8, 20, 19, 6];
scores.sort(Array.NUMERIC);
trace(scores);   // Displays: 2,5,6,8,10,14,19,20
```

فى بعض الاحيان تود ترتيب المصفوفة فى حالة اذا كانت تحتوى على عناصر فريدة غير متكررة لذلك يمكنك ان تستخدم *Array.UNIQUESORT*

مثال :

```
var ranking:Array = [2,5,6,3,1,1,4,8,7,10,9];
var sortedRanking:Object = ranking.sort(Array.UNIQUESORT);
trace(sortedRanking);   // Displays: 0
trace(ranking);  // Displays: 2,5,6,3,1,1,4,8,7,10,9
```

مثال على عمل تغيير عشوائي للترتيب :

```
var numbers:Array = new Array(  );
for(var i:int=0;i<20;i++) {
   numbers[i] = i;
}
numbers.sort(randomSort);
for(var i:int=0;i<numbers.length;i++) {
   trace(numbers[i]);
}
```

الـفـصـل الـثـالـث

الـمـعـامـلات Operators

انك تريد من برنامجك ان يفعل اشياء معينه عند تحقق شروط معينه ولكن كيف نصيغ هذه الاوامر هذا هو المهم انك ترى الشروط دائما وربما بدون ان تدري فأول شئ فعلته اليوم عند اتصالك بالانترنت هو ادخال كلمة السر – أليس كذلك – نعم هو كذلك ولا شئ غيره – ان برنامج الاتصال قد وضع شرط صحة كلمة السر لكي يمكنك من الدخول وربما اشياء اخرى مثل عدم انتهاء مدة الاشتراك وصحة اسم المستخدم وهكذا كلها شروط انها ايضا موضوعنا هذا اليوم ومعظم الشروط المستخدمه لقيمتين هي هل هما متساويتان ام احدهما اكبر او اصغر بالاضافة الى المعاملات البولينية Boolean operators

ومن درس منكم علم الجبر سوف يتذكرها بسرعة ام من نسى او لم يدرسها اصلا فالموضوع بسيط جدا لانك تمارسه في حياتك اليومية ربما دون ان تدري وهي

AND , OR , NOT

كمثال من الحياة اليومية :-

☞ **Ask Michael AND Nabil**

يجب ان تسأل مايكل و نبيل وليس واحد منهم فقط بل يجب ان يكون السؤال للاثنين

☞ **Ask Michae OR Nabil**

اي يجب ان تسأل مايكل او نبيل يكفي فقط واحد منهم

☞ **Ask Michael NOT Nabil**

اي يجب ان تسأل مايكل وليس نبيل فلو سألت نبيل فان الشرط غير متحقق

باختصار المعاملات تعتبر ضرورة بالنسبة لصياغة الشروط .

الجدول التالي يعبر عن طريقة صياغة المعاملات البولينية في الاكشن اسكربت :-

Operator	Description
+	الجمع
-	الطرح
!	ليس Not
++	لزيادة بواحدة و إسناد الناتج إلى المتغير
--	نقصان بواحدة و إسناد الناتج إلى المتغير
()	Grouping or function call
[]	Array element
.	Member access
++	Pre-increment
--	Pre-decrement
new	Allocation
delete / <=	Deallocation / اقل او يساوي
typeof / >	Type of object / اكبر من
void / >=	Undefined type / اكبر من او يساوي
== / *	التساوي / الضرب
!= / /	لا تساوي / القسمة
% / &	Bitwise AND / باقي القسمة
<< / ^	Bitwise shift left / Bitwise XOR
>> / |	Bitwise shift right / Bitwise OR
>>> / &&	Bitwise shift right (unsigned) / Logical AND
|| / <	أو / اقل من

?:	Conditional
=	Assignment
*=, /=, %=, +=, -=, &=, |=, ^=, <<=, >>=, >>>=	Compound assignments

تذكر جيدا خطأ من السهل ان تقع فيه ان الشروط غير التخصيص بمعنى انك عندما تخصص قيمة لمتغير فإنك لابد ان تستخدم علامة يساوي واحدة فقط مثل :

```
A=5;
B=A;
C=18;
```

اما في صياغة شرط فإنك لابد ان تضع علامتين اذا اشترطت التساوي مثل :-

```
if (A==B)
if (A==18)
```

ارجو ان تنتبه لذلك !!!!!!!!!!!!!!

يجب ان تعرف كيف تكتب المعادلات لانجاز الحسابات

هذا الموضوع بالذات تتفق فيه جميع لغات الكومبيوتر بلا استثناء وهناك بعض القواعد البسيطة التي يجب ان تعرفها مثل :-

١- لابد ان تحتوي المعادلة في طرفها الايسر على متغير واحد فقط واي عدد في الطرف الايمن

مثال :-

A = B + C * 5 –3

٢- جميع القيم الموجودة في الطرف الايمن تكون معلومة القيمة اي تم حسابها مثلا في معادلة سابقة

مثال :-

```
B = 5
C =  10
A = B + C * 5 –3
```

فقد تم تخصيص قيمة للمتغير B و للمتغير C

ثم دخل البرنامج الى المعادلة الاخيرة لحساب قيمة A وقد علم تماما قيم الطرف الايسر

اي قيم B و C

فاذا كتبت مايلي سيكون خطأ قاتل !!!!!!!!!!!

```
B = 5
A = B + C * 5 –3
C = 10
```

لان البرنامج عرف قيمة المتغير B ثم دخل الى المعادله ليحسب A وهو لا يعرف قيمة المتغير C حيث سيعلمها فيما بعد فرتب خطواتك لتكون منطقية ....

- **كل من درس مبادئ الرياضيات يعلم ان القسمة على الصفر خطأ فادح واذا حدثت سيتعطل البرنامج ويعطيك نظام التشغيل الرسالة المعروفة (هذا البرنامج قام بعملية ممنوعة)**

مثال

B = 7

C = 7

D = B-C

A = B/D

قام البرنامج بالقسمة على صفر حيث ان قيمة B = ٧ وC = ٧ وD = B – C اي ٧-٧ وتساوي بذلك صفر ثم تأتي مرحلة انهيار البرنامج عندما يقسم بي على دي وقيمتها صفر بعد الحسابات السابقة (احذر ذلك)

كل ماهو خطأ رياضي سيكون خطأ في البرنامج مثل حساب الجذر التربيعي لعدد اقل من الصفر وهكذا

- **حافظ دائما على ان يكون عدد الاقواس المفتوحة يساوي عدد الاقواس المغلقه**

اقصد عدد الاقواس ذات الاتجاه اليمين = عدد الاقواس ذات الاتجاه اليسار

لان الكومبيوتر عندما تفتح له قوس ايسر معناها بدء عملية حسابية منفصلة حتى لوكانت داخل معادلة واحدة

فأذا لم تغلق بالقوس الايمن فمعنى ذلك عملية لم تغلق وتنهار المعادلة وبالتالي البرنامج.

مثال لاحظ فيه ان عدد الاقواس متساوي

A = (A+B) * (C – D) – (Q * M)

<u>عملية باقي القسمة :-</u>

نحن نعرف العمليات الحسابية الأربعة الجمع و الضرب و القسمة و قد اعتدنا عليها و لكن في بعض لغات البرمجة هناك عملية خامسة تضاف عليها و هي باقي القسمة حيث تضع بين أي عددين و تدل على باقي قسمة الأول على الثاني مثل :-

z = x % y ;

<u>عملية الزيادة بواحد أو النقصان بواحد(++ ، --) :-</u>

و تتلخص مهمة كلا من هذه العمليات على زيادة واحد على قيمة أي متغير و ثم تسند الناتج له أي إذا كان هناك متغير يساوي ٥ ووضعنا هذه العملية قبله أو بعده فسوف يصبح ٦ و لاحظ هنا أني قلت قبله أو بعد حيث هناك فرق بين الحالتين .

اين المثال ؟؟؟؟؟؟؟؟؟؟؟؟؟؟؟؟؟؟؟؟؟؟

من الواضح أن المتغير y سوف يزيد بواحد أي تصبح قيمته ٦ و من ثم تسند هذه القيمة إلى المتغير z

لكن أنظر إلى الحالة التالية و لاحظ الفرق في المثال التالي :-

```
y = 5;

z = y++ ;
```

أما في هذه الحالة فسوف تسند قيمة **y** الأصلية إلى **z** و هي القيمة ٥ و من ثم تزاد قيمة **y** بواحدة .



Pre-Increment and Post-Increment (++)

قمت بتعريف متغير اعطيتة القيمة ١٠ و هذا السطر لن يتم تفيذة من البرنامج لانه تعليق //

```
myNumber = 10;
trace(++myNumber);
trace(myNumber++);
trace(myNumber);
```

الناتج
١١
١١
١٢

Pre-Decrement and Post-Decrement (--)

```
myNumber = 10;
result = --myNumber * 2;
trace(result);
myNumber = 10;
result = myNumber-- * 2;
trace(result);
```

الناتج
١٨
٢٠

مثال على عمل دالة لجمع رقمين :-

قم بعمل ٣ مربع نص و ليكن اسمهم **input1** و **input2** و **total** و زر اسمة btAdd
ثم ضع الكود التالى فى الفريم الاول

```
btAdd.onRelease = function(){
    var theTotal:Number = Number(input1.text)+Number(input2.text);
    total.text = String(theTotal);
}
```

الـــفـــصـــل الـــرابـــع

جـمـل الـتـحـكـّم فـي الـمـسـار Flow- control statement

إنّ البرمجة أعمق من أن تكون مجرّد تعريف متغيّرات.. إنّها تفكير منطقيّ يعتمد على حساب كل الاحتمالات، لاتخاذ الأفعال المناسبة لكل احتمال.. لهذا فلا بد أن توجد طرق نتحكّم بها فيما ينفّذ ومتى ينفّذ من البرنامج.

جملة الشرط If...:

تستطيع أن تختبر حدوث شرط معيّن، فإذا كان صحيحا يتمّ تنفيذ مقطع الشرط، وإن كان خاطئا يقفز التنفيذ إلى جملة نهاية الشرط .

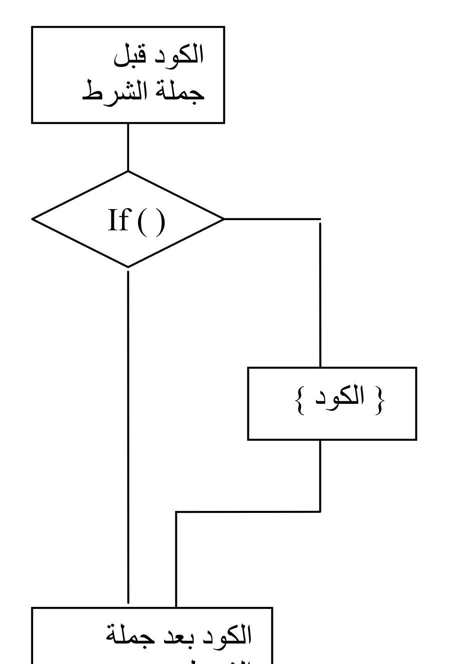
ولكن ما هو الشرط؟؟؟؟؟؟؟؟؟؟؟؟؟؟؟

الشرط **هو افتراض معين يتوقف عليه عمليات اخرى فمثلاً تريد ان تضع شرط الا يدخل رقم موظف اكبر من الف لان عدد موظفين الشركة لايزيدون عن الف وبالتالي اذا ادخل من يعمل على البرنامج رقم موظف اكبر من الف يقوم البرنامج باصدار رسالة تفيد بذلك و هكذا لها حالات كثيرة حسب فكرة البرنامج**

المهم انه تعبير يعطي نتيجة منطقيّة (True أو False)، مثل:

```
myNumber = 10;
if(myNumber < 20){
   trace("myNumber is less than 20");
}
```

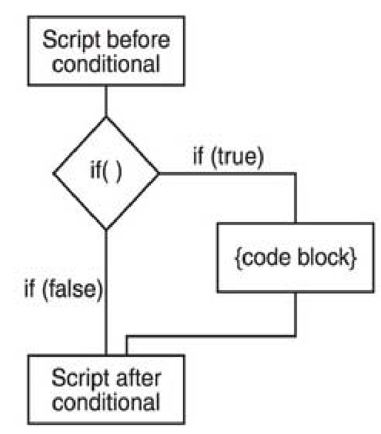
شرح المثال : لقد قمنا بتعريف متغير myNumber و اسندنا له القيمة ١٠ و قمنا بأختبار قيمة هذا المتغير اذا كانت اقل من ٢٠ فسوف نقوم بتنفيذ الكود الموجود بين القوسين { } و هو جملة trace اما اذا كانت قيمة المتغير اكبر من ٢٠ فلن يحدث شيء من الكود الموجود بين الاقواس { }



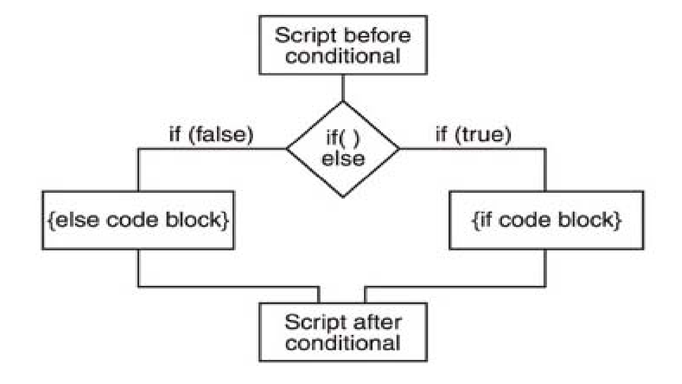
جملة الشرط else...If:

مثال :

```
myNumber = 10;
if(myNumber > 20){
   trace("myNumber is greater than 20);
}
Else{
   trace("myNumber is less than or equal to 20);
}
```



شرح المثال : لقد قمنا بتعريف متغير **myNumber** و اسندنا له القيمة ١٠ و قمنا بأختبار قيمة هذا المتغير اذا كانت اكبر من ٢٠ فسوف نقوم بتنفيذ الكود الموجود بين القوسين **{ }** و هو جملة **trace** اما اذاقمنا بتغيير قيمة المتغيرلتصبح ٥٠ فسيتم تنفيذ الكود الموجود بعد كلمة **else**

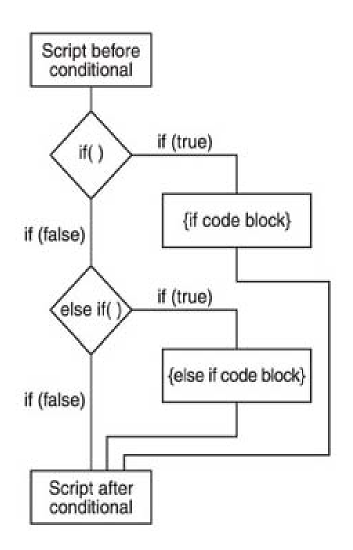


جملة الشرط if...Else :-

يمكن دمج جملتى **if** من خلال استخدام التعبير **else if**

```
myNumber = 10;
if(myNumber < 20){
trace("myNumber is less than 20");
}

else if(myNumber < 50){
trace("myNumber is less than 50 but greater than or equal to 20");
}
```



استخدام if مع معاملات اخرى مثل and او or:-

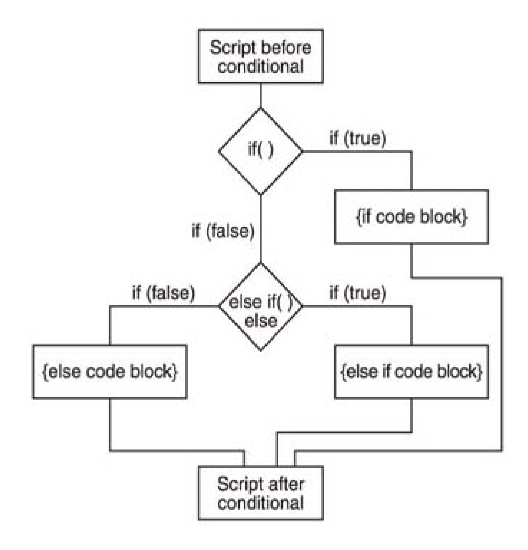
```
on (press) {
  if ((a == 7) and (b == 15)) {
   gotoAndPlay(20);
}
}
```

مثال اخر:-

```
on (press) {
  if ((a == 7) or (b == 15)) {
   gotoAndPlay(20);
}
}
```

كما يمكن دمج اكثر من تعبير فى جملة if كما يلى :-

```
myNumber = 10;
if(myNumber < 20){
   trace("myNumber is less than 20");
}
else if(myNumber < 50){
   trace("myNumber is less than 50 but greater than or equal to 20");
}
else{
   trace("myNumber is greater than or equal to 50");
}
```



الدالة الشرطية جملة Case .... switch:-

عندما يكون لدينا عدة خيارات و نكون نريد أن نخرج بواحد منهم و هو الذى نريده من بين الخيارات والذى سوف نخرج به سوف يحدده المتغير الذى سوف ندخله و الذى سوف يتفق مع واحدة من هذه الخيارات و يحققه ..

switch (*المتغير*){
case *الاحتمال الاول*:
المطلوب لهذا الاحتمال
case *الاحتمال الثانى*:
المطلوب لهذا الاحتمال
فى حالة عدم تطابق اى حالة يتم تنفيذ ال
...
[default]
}

The switch Statement

```
switch (expression){
   case caseClause1:
      code block
   case caseClause2:
      code block
   ...
   [default]
}
```

مثال:-

فى المثال التالى عرفنا متغيرx و قيمته ١٠ وهناك حالتين لهما نفس قيمة شرط الحالة و هو ١٠ لذلك سوف تجد الناتج دائما هو الحالة الاولى case 1 و لن يطبع ابدا الحالة الثانية.

```
x = 10;
switch( x ){
```

```
    case 10:
        trace("case 1");
        break;
    case 10:
        trace("case 2");
        break;
}
```

مثال اخر:-

```
exp = "hello";
switch( exp ){
    case "hello":
        trace("case 1");
    case "hi":
        trace("case 2");
        break;
}
```

الناتج

```
case 1
```

مثال أخر يوضح التشابة بين فكرة استخدام if للتحكم فى البرنامج و استخدام switch :-

```
switch( exp ){
    case 1:
        //do task a
    case 2:
        //do task b
    case 3:
        //do task c
}
```

يمكن عمل نفس الكود السابق باستخدام if هكذا :-

```
if( x == 3 ){
    //do task c
}
else if( x == 2 ){
    //do task b
    //do task c
}
else if( x == 1 ){
    //do task a
    //do task b
    //do task c
}
```

مثال اخير يوضح جملة switch :-

```
switch( user_command_string ){
    case "move north":
    case "go north":
    case "north":
    case "n":
        trace("you have moved north");
        break;
```

```
  case "move south":
  case "go south":
  case "south":
  case "s":
    trace("you have moved south");
    break;
  default:
    trace("I'm sorry, I don't understand "+ user_command_string);
}
```

الخلاصة : اننا ندخل المتغير فى جملة (switch) للمقارنة مع الخيارات الموجودة بداخلها و عند مطابقة المتغير مع احدى الخيارات تصبح النتيجة (true) وسوف تنفذ الجملة المتعلقة بهذه المطابقة مع العلم ان واحد فقط من هذه الخيارات يعطى (true) و البقية (false) و عندما تكون كل الخيارات ليست مطابقة سوف ينفذ ما بداخل الـ(default) .

الـفـصـل الـخـامـس

تـراكـيـب الـتـكـرار Loop Structures

قدرة الكمبيوتر على تكرار أي جزء من الكود - خاصّةً مع سرعته الفائقة - هي ما تجعله مريحا جدا للبشر، ليحمل عنهم عناء الرتابة والبطء والملل

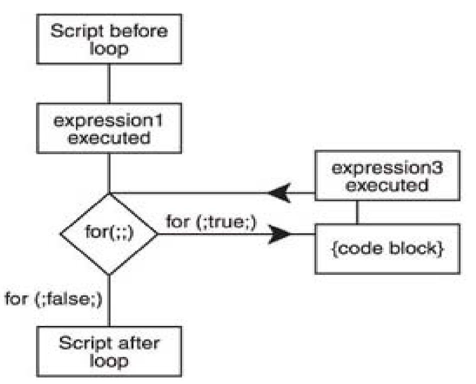
جملة التكرار "من إلى" ...For:

الدوران او التكرار او عمل looping من الاوامر الاساسية في جميع لغات البرمجة فمثلا اذا كان البرنامج سيدخل اسماء الف موظف هل تعتقد انك ستكتب الف امر لادخال هذه الاسماء بالطبع ستكون حماقه ولكن لو وضعنا امر واحد فقط لادخال اسم الموظف وطلبنا من البرنامج الدوران الف مرة حول هذا الامر بالطبع سيكون شئ جميل ان يدخل الف بيان بمجهود بسيط نتيجة تسهيل اعطته لغة البرمجة

مثال

```
for(i = 0;i < 1000; i++){
  trace(i);
```

}



ياله من شئ جميل فعلاً طلبنا من الكومبيوتر انشاء عداد يعد من صفر الى ١٠ ورمزنا لها بالرمز i ثم داخل العداد طلبنا منه ادخال الموظف رقم i وهو عداد متغير حتى يكتمل العداد بوصوله للالف ويكون قد تم تنفيذ الامر معه الف مرة بأدنى مجهود وكلما تغير العد من ١ الى ٢ الى ١٠٠٠ تغير معه رقم الموظف بنفس الطريقة

**** الجدول التالي يوضح صيغ التكرار المختلفة الخاصة بـFor وتفسيرها.............**

الصيغة	التفسير
for	امر اللغة لعملية التكرار
(	قوس مفتوح يوضح بداخله بارامترات الامر
i=0;	المتغير = رقم بداية الحلقات
i<1000;	شرط نهاية الحلقات
i++	المتغير يزيد بمقدار واحد مع بداية كل حلقة – لاتضع بعده فاصلة منقوطة
)	قوس نهاية بارامترات الامر – لاتضع بعده فاصلة منقوطة
{	قوس بداية بلوك الاوامر المطلوب تكرارها
بداخل اقواس البلوك توضع الاوامر المطلوب تكرارها	
}	قوس نهاية بلوك الاوامر المطلوب تكرارها

بداخل بلوك الاوامر تم تنفيذ الامر

وهو امر يقوم بطبع قيمة x التي تتغير في كل مرة ابتداء من صفر حسب ماذكرت ان

i=0 وتزيد في كل مرة بمقدار ١ حسبما ذكرت ان i++ وذلك حتى يصل الى ٩٩ حسبما ذكرت ان i<100

وبالتالي ستكون مخرجات البرنامج كما يلي

0

1

2

3

4

.

.

.

99

لنستفيد اكثر من قوة الحلقات التكرارية وايضا نرى امكانات اخرى لها بوضع حلقة داخل حلقة كما يلى في البرنامج الذي يطبع جدول الضرب من جدول واحد حتى جدول ١٢

```
on(press){
for(var x:Number=1;x<13;x++) {
  for(var y:Number=1;y<13;y++) {
   var z:Number = x * y
         trace(x+"*"+y+"="+z)
  }}}
```

اعلنا عن ثلاث متغيرات لاعداد صحيحة الاول للحلقة الاولى والثاني للحلقة الثانية والثالث لاحتواء حاصل الضرب وهو الهدف من البرنامج.

بنفس ماسبق شرحه وضعنا حلقة تكرارية تبدء من رقم واحد وتنتهي برقم ١٢ وتزيد بمقدار واحد واوامر هذه الحلقة مكتوبة للتوضيح بخلفية صفراء حيث بدأت الحلقة بالصيغة السابق شرحها ثم قوس بداية البلوك وقوس نهاية البلوك حيث يوضع مابين القوسين سلسلة اوامر لتنفيذها كما سبق واتفقنا عليه

```
for(var x:Number =1;x<13;x++)
{
```

اوامر مطلوب تنفيذها

```
{
```

وبداخل اقواس البلوك للحلقة الاولى مطلوب وضع اوامر للتنفيذ فكانت اوامر التنفيذ داخل البلوك عبارة عن حلقة اخرى ، وما المانع فالحلقات وغيرها هي نفسها اوامر والحلقه الاخرى كانت عبارة عن عداد يعد من الرقم ١ الى الرقم ١٢ وهي ايضا تحتاج الى صيغة معينة واقواس بلوك للتنفيذ وهي موضوعة في البرنامج بخلفية خضراء كما يلي :-

```
for(var y:Number=1;y<13;y++)  {
        var z:Number = x * y;

trace(x+"*"+y+"="+z)

}
```

وبداخل بلوك الاوامر للحلقة الثانية وضعنا عدة اوامر لتفى بالغرض الذي من اجله تم عمل البرنامج وهو جدول الضرب – عبارة عن معادلة لحساب حاصل ضرب المتغيرين x و y

ويوضع الناتج في المتغير z كما يلي :-

```
z = x * y;
```

يبدء البرنامج بالدخول الى الحلقه الاولى ويخصص للمتغير X القيمة واحد حيث انها قيمة عداد البداية ثم يدخل الى داخل اقواس بلوكه لتنفيذ ما بداخله فيجد حلقه اخرى فيبدء بتنفيذها وطلبت الحلقة ان يكون متغيرها Y يبدء بالقيمة واحد وينتهي بالقيمة ١٢ ثم تدخل الحلقة الثانية الى بلوك اوامرها فتجد معادلة يضرب فيها قيمة x التي هي واحد الآن وتتغير قيمة y فيها ثم طبع النتائج وتظل الحلقة الداخلية تنفذ بتغير Y مع ثبات X بالطبع وتكون النتائج كما يلى :-

1 * 1 = 1

1 * 2 = 2

1 * 3 = 3

.

.

.

1 * 12 = 12

وعند وصول نهاية عداد الحلقة الداخلية الى ١٢ وهو نهاية الحلقة ينتهي تنفيذ الحلقة فيستمر البرنامج فيجد امامه قوس نهاية الحلقه الخارجية ذو الخلفية الصفراء فيعود الى الحلقة الخارجيه حيث تزيد قيمة X بمقدار واحد وتصبح قيمتها = ٢ فيدخل الى بلوك اوامره كما سبق فيجد حلقة اخرى تبدء من واحد وتنتهي بـ ١٢ وبداخلها معادلة وامر طبع فيكون التنفيذ كما يلي :-

2 * 1 = 2

2 * 2 = 4

2 * 3 = 6

.

.

.

2 * 12 = 24

و هكذا يستمر التنفيذ الى ان تصل الحلقة الخارجية الى نهايتها بالرقم ١٢ واثناء ذلك تنفذ الحلقة الداخلية من بدايتها الى نهايتها اي من ١ الى ١٢ وتطبق المعادلة وامر الطبع ليكون اخر تنفيذ كما يلي :-

12 * 1 = 12

12 * 2 = 24

12 * 3 = 36

.

.

.

12 * 12 = 144

مثال اخر لعمل ١٠ نسخ من نفس الـ movieClip لاحظ انه تم تسميتةmyClip_mc

```
for (var i:Number=0; i<=10 ; ++i) {
  myClip_mc.duplicateMovieClip("myClip_mc" + i, i);
}
```

جملة التكرار The while Loop

فى هذه الطريقة يستمر بتنفيذ ما بداخل جملة التكرار ما دام الشرط متحقق فى كل مرة تريد فيها الدخول سوف يتحقق من الشرط اولا فاذا تحقق تقوم بالدخول الى داخل الجملة و تنفذ ما بداخلها الى ان يفشل و يخرج من جملة التكرار و لكن اذا كان هناك اوامر خارج جملة التكرار اى بعد جملة while سوف ينفذها .

لكن لاحظ ان جملة (while) يجب ان تحتوى على ما يلى:

١- متغير نضعه بالشرط لكى نتحقق من صحة الشرط .

٢- يجب وضع قيمة ابتدائية لهذا المتغير قبل جملة الـ(while) .

٣- يجب ان نذكر هذا المتغير و مقدار زيادته بداخل جملة الشرط سواء قبل تنفيذ الجملة التى بداخل جملة التكرار او بعدها .

مثال :-

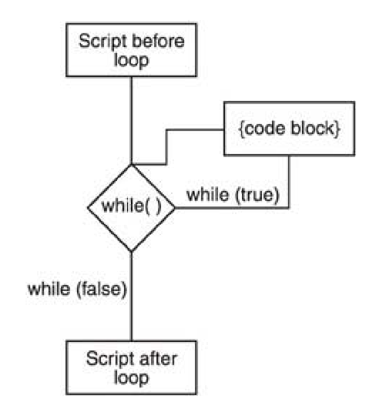
```
x = 0;
while( x < 100 ){
   trace(myNumber);
   x++;
}
```

** الجدول التالى يوضح صيغ عمليات التكرار المختلفة وتفسيرها ....

الصيغة	التفسير
while	امر اللغة لعملية التكرار
)	قوس مفتوح يوضح بداخله بارامترات الامر

X<100	المتغير مع شرط لنهاية الحلقة
)	قوس نهاية بارامترات الامر – لاتضع بعده فاصلة منقوطة
{	قوس بداية بلوك الاوامر المطلوب تكرارها
بداخل اقواس البلوك توضع الاوامر المطلوب تكرارها	
x++;	ولا تنسى عداد الزياده او النقصان ليتحقق الشرط لانهاء تنفيذ الحلقة
}	قوس نهاية بلوك الاوامر المطلوب تكرارها

وبعد الاعلان عن المتغير X خلاصة شرح الامر السابق (أعد تنفيذ ما بداخل البلوك طالما X او المتغير اقل من ١٠٠ ثم اقواس بلوك تضع ماشئت بداخله من اوامر وقوس نهاية البلوك ويزيد معنا فقط عداد للمتغير ليزيده بالمقدار الذي تريده ويوضع في اي مكان داخل الحلقة او حسب افكارك عن البرنامج المهم لاتنساه والا سوف يدور البرنامج داخل الحلقة الى الابد حيث ان شرط نهايتها ان تزيد X عن ١٠٠ وطالما لم تضع عداد زياده للمتغير فلن يتحقق الشرط وبالتالى لن تنتهي الحلقة الى الابد ويظل يعمل الكومبيوتر بلا نهاية للبرنامج ويميز العداد بالخلفية الصفراء واليك جدول صياغة الامر الذي تعودنا عليه.



The do while Loop حلقة التكرار

فى هذه الحالة يستمر تنفيذ ما بداخل الحلقة ما دام الشرط متحقق و هنا سوف يدخل الى داخل الحلقة و من ثم ينفذ الامر الذى بداخلها و بعد تنفيذها ينتقل ليتحقق من الشرط فاذا تحقق يعود مرة اخرى و اذا لم يتحقق يخرج من حلقة التكرار و لن يعود لها .

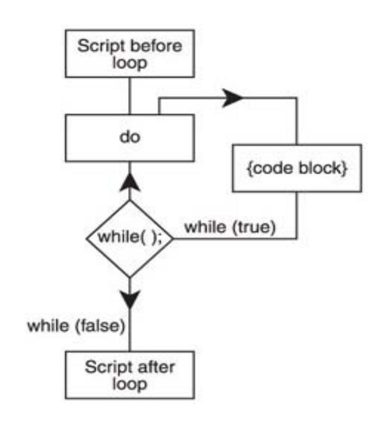
لكن لاحظ ان جملة (do while) يجب ان تحتوى على ما يلى:

١- متغير نضعه بالشرط لكى نتحقق من صحة الشرط .

٢- يجب وضع قيمة ابتدائية لهذا المتغير قبل جملة الـ(do while) .

٣- يجب ان نذكر هذا المتغير و مقدار زيادته بداخل حلقة الشرط سواءا قبل تنفيذ الجملة التى بداخل حلقة التكرار او بعدها .

```
myNumber = 99;
do{
   trace(myNumber);
}while(myNumber++ < 10);
```



قد نتسائل هنا ما الفرق بين **while** و جملة **do while** ؟؟؟؟
فى **(while)** نتحقق من الشرط قبل الدخول الى الحلقة اى اننا لا ننفذ اى شئ بداخلها ما دام الشرط لم يتحقق و هذا امر طبيعى لاننا لم ندخل الى الحلقة اصلا فكيف نعرف ما بداخلها و ننفذه أما فى **(do while)** كنا ندخل الى الحلقة و ننفذ امر ثم نفحص الشرط و لكن بعد ان نكون قد نفذنا هذا الامر و يجب التنبيه هنا فى حالة عدم تحقق الشرط لن نعود مرة اخرى الى الـ**(do)** اذن الفرق هو ان بالـ **(do while)** ينفذ على الاقل امر واحد فى داخل حلقة التكرار حتى لو كان الشرط غير متحقق على العكس الـ**(while)** الذى لا ينفذ اى امر مادام الشرط غير متحقق .

تداخل التكرار Nested Loops

```
var i:Number = 0;
while (++i <= 10) {
  var j:Number = 0;
  while (++j <= 10) {
    // perform these actions
  }
}
```

الان اعتقد ان من الممكن دمج بعض الجمل و التعبيرات لعمل مثال نستخدم فية المصفوفات و التكرار ...

```
// عرف مصفوفة جديدة
var myArr:Array = new Array();
// value1 قمنا بأسناد قيم لعناصر المصفوفة الاول قيمتة
myArr[1] = "value1";
myArr[0] = "value0";
// للمرور على العناصر الموجودة داخل المصفوفة و معرفة قيمتها سوف استخدم التكرار
var i:String;
for (i in myArr) {
  // طباعة القيم
  trace("key: " + i + ", value: " + myArr[i]);
}
```

الناتج

```
key: 0, value: value0
key: 1, value: value1
```

الـفـصـل الـســادس

functions and methods

الـدوال و الـخـصـائـص

الدوال functions هي شبه برنامج صغير اذا كان عندك سلسله عمليات متشابهه سوف تجريها داخل البرنامج كثيرا وتكررها كثيرا فلا داعي لاعادة كتابتها كل مرة بل يكفيك ان تكتبها مرة واحدة وتطلق عليها اسم وكلما ذكرت هذا الاسم داخل البرنامج تتم تلك العمليات ويصبح هذا الاسم كأنه من اوامر اللغة ، ايضا من فوائدها :-

- تقسيم البرنامج الى اجزاء صغيرة تستدعى وقت اللزوم .
- تقسيم البرنامج الى اجزاء يمكن اختبارها منفصلة لسرعة تحديد الخلل بالبرنامج .
- تقسيم العمل بين المبرمجين عند العمل في مشروع جماعي لانتاج برنامج كبير .
- تبادل الخبرات بين مطوري البرامج بنشر اجزاء يستخدمها الآخرون في برامجهم .

مثال اخر :-

لنفرض ان هناك اوامر لرسم جدول بأبعاد ومساحة معينة وتحتاج الى رسمة كثيرا فليس معنى هذا انك كل مرة تريد رسم الجدول ستعيد كتابة اوامره الكثيرة في كل مكان داخل البرنامج فقط جمع اوامر رسمه في دالة واعطية اسم وكلما ذكرت الاسم يتم رسم الجدول و نقوم بأعطاء هذة الدالة معطيات خاصة بها عند استدعائها انه شئ رائع اليس كذلك انا اظن ذلك.

الشكل الاساسى لكتابة الدالة هو:-

```
function myFunction (parameter1:DataType,parameter2:DataType,etc.) {
  //actions go here;
}
```

حيث myFunction اسم الدالة
parameter1 المعطى الاول للدالة و نوعة
و عندما نريد استدعاء هذه الدالة نقوم بكتابة الكود التالى :

```
myFunction(parameter1, parameter2);
```

لتطبيق ذلك دعنا نحضر البرنامج الذي تناولناه سابقا الخاص بجدول الضرب لنعطيه مزيد من القوة و التطوير باستخدام الدوال ،

ودعنا ننتهز فرصة الدوال لاعطاءمزيد من التحسينات على البرنامج حيث ان البرنامج ينتج فقط جدول الضرب من جدول ١ الى جدول ١٢ فلماذا لانجعلها دالة عامة تنتج مايحلو لنا من جداول ضرب من المؤكد انك تتفق معي في ذلك وبذلك نستفيد من قوة الدوال وتصبح دالة عامة غير موجهه لمهمة ثابته فنحن نريد ان نحدد نحن بداية الجداول ونهايتها فتارة تطبع من جدول ٧ الى جدول ٢٠ وتارة تطبع من جدول ١٧ الى ٤٠ وهكذا اليس هذا افضل سوف نمرر للدالة قيمة البداية وقيمة النهاية مع اجراء التعديلات اللازمة لذلك.

الصيغة العامة لكتابة الدوال :-

```
function اسم الدالة(المتغيرات التى سوف ترسل للدالة)
{
 بداية الدالة العمليات التى تتم بداخل الدالة
Return النتيجة المعادة و;
}
```

```
on(press)
{
  function myFunction1(s1:Number,e1:Number)
  {for(var x:Number=s1;x<e1+1;x++)
{
  for(var y:Number=s1;y<e1+1;y++)
 {
   var z:Number = x * y
          trace(x+"*"+y+"="+z)
  }}}
//للأستدعاء الدالة نقوم بكتابة اسم الدالة و اعطائها قيمة المتغيرات بين القوسين
  myFunction1(1,5);
    }
```

لتسهيل الامر عليك وضعت لك لون اصفر على التعديلات التي اجريتها للوصول الى الهدف المنشود لقد وضعنا في اقواس **Prototype** اعلان عن تمرير متغيرين رقمين الى الدالة (**s1:Number,e1:Number**)
حيث **s1** هو الرقم الذى سيبدأ منه الجدول و المتغير **e1** هو الرقم الذى سينتهى فيه الجدول بعد ان ادخلت قيمة البداية والنهاية فكما ذكرنا تم تعديل الدالة لتمرير متغيرين لها يتغيران مع كل ادخال والآن لننادي الدالة ممرين لها القيمتين الذين ادخلهما مستخدم البرنامج

```
  myFunction1(1,5);
```

مثال: دالة لحساب الوزن مقسوما على ٦,٠٤

```
function convertToMoonWeight (myWeight:Number){
  var weightOnMoon:Number = myWeight/6.04;
}
```

و لكى اقوم بأستدعاء الدالة السابقة نكتب التالى :-

```
convertToMoonWeight(165);
```

مثال اخر :-

```
function openWindow(url:String, window:String){
getURL(url, window);
}
```

لأستدعائها نكتب الكود التالى :-

```
openWindow("http://www.yahoo.com", "_blank");
```

مثال اخر :-

```
on (press) {
          function myFunction(num) {
    var newNum = num + 3;
    return newNum;
```

```
}
  trace(myFunction(7));
trace(myFunction(13));
trace(myFunction(2));
}
```

الناتج

10
16
5

★★★★★★★★★★★★★★★★★★★★★★★★

الــفــصــل الــســابــع

الــرســم بــواســطــة الاكــشــن اســكــريــت

يمكننا الرسم باستخدام الدالة lineStyle() لكن هذة الدالة لها معطيات منها نوع الخط و درجة وضوحه و لونه

```
path.lineStyle(thickness, color, alpha)
```

مثال :-

```
_root.myClip_mc.lineStyle(10, 0x009900, 100);
```

للتحرك للمكان الذى سيبدأ منه الرسم نستخدم الدالة moveTo()

```
path.moveTo(x, y);
```

مثال :-

```
_root.myClip_mc.lineStyle(10,0x009900,100);
_root.myClip_mc.moveTo(100,100);
```

لنبدا الرسم من نقطة x الى نقطة y نستخدم الدالة lineTo()

مثال :-

```
_root.createEmptyMovieClip("canvas_mc",1);
_root.canvas_mc.lineStyle(2,0x009900,100);
_root.canvas_mc.moveTo(100,100);
_root.canvas_mc.lineTo(200,150);
```

مثال لعمل خط منحنى :-

```
this.createEmptyMovieClip("myMovieClip_mc", 1);
myMovieClip_mc.lineStyle(1, 0x000000, 100);
myMovieClip_mc.curveTo(0, 100, 100, 100);
```

مثال اخر لعمل مستطيل:-

```
_root.createEmptyMovieClip("rectangle_mc", 1);
rectangle_mc.lineStyle(1, 0x000000, 100);
// رسم الخطوط الاربعة المكونة للمستطيل
rectangle_mc.lineTo(100,  0);
rectangle_mc.lineTo(100, 50);
rectangle_mc.lineTo(  0, 50);
rectangle_mc.lineTo(  0,  0);
```

مثال لعمل دائرة:-

```
MovieClip.prototype.drawCircle = function (radius, x, y)
{
var angleDelta = Math.PI / 4;
  var ctrlDist = radius/Math.cos(angleDelta/2);
```

```
  var angle = 0;
  var rx, ry, ax, ay;
  this.moveTo(x + radius, y);
  for (var i = 0; i < 8; i++) {
    angle += angleDelta;
    rx = x + Math.cos(angle-(angleDelta/2))*(ctrlDist);
    ry = y + Math.sin(angle-(angleDelta/2))*(ctrlDist);
    ax = x + Math.cos(angle)*radius;
    ay = y + Math.sin(angle)*radius;
    this.curveTo(rx, ry, ax, ay);
  }
}
var ctrlDist = radius/Math.cos(angleDelta/2);
rx = x + Math.cos(angle-(angleDelta/2))*(ctrlDist);
ry = y + Math.sin(angle-(angleDelta/2))*(ctrlDist);
ax = x + Math.cos(angle)*radius;
ay = y + Math.sin(angle)*radius;
this.createEmptyMovieClip("circle_mc", 1);
circle_mc.lineStyle(1, 0x000000, 100);
circle_mc.drawCircle(100, 50, 75);
circle_mc.drawCircle(65);
```

مثال لعمل مربع و تلوينه باللون الاحمر :-

قمنا بعمل movieclip فارغ و قمنا بتسميته box_mc بالكود//

```
_root.createEmptyMovieClip("box_mc",1);
with (_root.box_mc) {
  lineStyle(0,0x000000,100);
  beginFill(0x990000,100);
  moveTo(0,0);
  lineTo(100,0);
  lineTo(100,100);
  lineTo(0,100);
  lineTo(0,0);
  endFill();
}
```

مثال اخر لعمل مستطيل به تدريج لونى :-

```
_root.createEmptyMovieClip("holder_mc", 1);
with (_root.holder_mc) {
  lineStyle(0, 0x000000, 0);
  rotation = 90 * (Math.PI/180);
  colors = [ 0x6666FF, 0xFF6600 ];
  alphas = [ 100, 100 ];
  ratios = [ 0, 255 ];
```

```
  matrix = {matrixType:"box", x:0, y:150, w:200, h:100, r:rotation };
  beginGradientFill( "linear", colors, alphas, ratios, matrix );
  moveTo(0,0);
  lineTo(550,0);
  lineTo(550,300);
  lineTo(0,300);
  lineTo(0,0);
  endFill();
}
```

ملحوظة :- بعد كتابة الكود قم بالضغعط على مفتاحى Ctrl + enter لمشاهدة الشكل الذى قمت برسمه .

رسم فتاة بالاكشن اسكربت

```
_root.createEmptyMovieClip("girl", 1);
with (_root.girl) {
lineStyle(0, 0x000000, 100);
beginFill(0x000000, 100);
moveTo(138, 164);
curveTo(119, 160, 98, 165);
curveTo(120, 155, 138, 164);
moveTo(156, 166);
curveTo(166, 162, 174, 164);
lineTo(174, 161);
curveTo(164, 160, 156, 166);
endFill();
lineStyle(2, 0x000000, 100);
moveTo(133, 178);
curveTo(116, 166, 104, 183);
moveTo(132, 181);
curveTo(116, 169, 104, 184);
moveTo(132, 184);
curveTo(117, 170, 104, 184);
moveTo(128, 193);
curveTo(119, 197, 109, 193);
moveTo(126, 179);
curveTo(130, 191, 121, 195);
moveTo(117, 195);
curveTo(106, 184, 116, 175);
lineStyle(0, 0x000000, 100);
moveTo(115, 179);
curveTo(118, 180, 118, 183);
curveTo(116, 185, 113, 183);
curveTo(114, 190, 119, 191);
curveTo(125, 190, 125, 185);
curveTo(124, 177, 115, 179);
lineStyle(2, 0x000000, 100);
moveTo(151, 185);
curveTo(151, 177, 160, 173);
curveTo(168, 171, 173, 176);
moveTo(154, 180);
```

```
curveTo(162, 170, 171, 178);
moveTo(163, 174);
curveTo(170, 175, 170, 182);
moveTo(155, 192);
curveTo(163, 195, 167, 191);
moveTo(160, 175);
curveTo(153, 181, 155, 189);
curveTo(157, 192, 161, 192);
curveTo(171, 188, 163, 176);
lineStyle(0, 0x000000, 100);
moveTo(160, 180);
curveTo(162, 182, 161, 183);
curveTo(159, 184, 158, 182);
curveTo(156, 187, 160, 189);
curveTo(164, 189, 164, 185);
curveTo(164, 180, 160, 180);
lineStyle(2, 0x000000, 100);
moveTo(148, 185);
curveTo(147, 195, 153, 203);
curveTo(147, 207, 147, 209);
moveTo(130, 219);
lineTo(152, 218);
lineStyle(3, 0x000000, 40);
moveTo(133, 225);
lineTo(143, 225);
lineStyle(1, 0x000000, 100);
moveTo(174, 145);
curveTo(176, 157, 175, 172);
curveTo(173, 181, 172, 186);
lineTo(169, 212);
curveTo(169, 217, 164, 224);
lineTo(148, 242);
curveTo(144, 246, 135, 246);
curveTo(112, 242, 96, 230);
moveTo(71, 168);
curveTo(68, 164, 63, 164);
curveTo(58, 164, 55, 172);
curveTo(56, 191, 68, 204);
lineTo(74, 206);
lineStyle(1, 0x000000, 60);
moveTo(70, 174);
curveTo(67, 170, 63, 169);
curveTo(54, 174, 62, 185);
curveTo(68, 200, 72, 192);
moveTo(71, 180);
curveTo(71, 177, 68, 177);
curveTo(65, 177, 65, 180);
curveTo(66, 185, 64, 189);
```

```
curveTo(70, 200, 72, 189);
curveTo(67, 184, 71, 181);
lineStyle(1, 0x000000, 60);
moveTo(47, 151);
curveTo(55, 93, 113, 85);
lineTo(115, 87);
lineTo(116, 85);
curveTo(170, 92, 195, 130);
curveTo(207, 158, 194, 164);
curveTo(207, 152, 185, 123);
curveTo(203, 150, 189, 169);
curveTo(193, 150, 186, 138);
curveTo(197, 169, 179, 179);
curveTo(190, 174, 184, 145);
moveTo(176, 127);
curveTo(190, 165, 178, 174);
curveTo(184, 159, 178, 137);
curveTo(182, 156, 173, 169);
curveTo(179, 155, 175, 139);
moveTo(171, 123);
curveTo(181, 153, 164, 167);
curveTo(176, 146, 168, 127);
curveTo(175, 149, 159, 160);
curveTo(171, 141, 166, 129);
moveTo(161, 113);
curveTo(175, 141, 144, 164);
curveTo(161, 151, 161, 128);
moveTo(158, 127);
curveTo(159, 151, 143, 164);
curveTo(157, 147, 155, 119);
curveTo(152, 144, 120, 157);
curveTo(157, 136, 152, 106);
moveTo(150, 115);
curveTo(146, 139, 130, 150);
curveTo(140, 140, 147, 117);
curveTo(132, 149, 113, 153);
moveTo(145, 121);
curveTo(139, 132, 126, 141);
curveTo(116, 154, 97, 160);
moveTo(118, 141);
curveTo(103, 156, 98, 156);
moveTo(118, 141);
curveTo(105, 149, 98, 155);
curveTo(87, 162, 70, 163);
moveTo(101, 156);
curveTo(94, 163, 72, 163);
moveTo(71, 164);
curveTo(72, 222, 105, 318);
```

```
moveTo(74, 164);
curveTo(74, 185, 78, 205);
moveTo(77, 164);
curveTo(75, 194, 88, 241);
moveTo(80, 165);
curveTo(90, 245, 101, 292);
moveTo(72, 187);
curveTo(73, 225, 80, 250);
moveTo(82, 165);
curveTo(82, 210, 103, 259);
moveTo(85, 164);
lineTo(84, 193);
moveTo(90, 162);
lineTo(85, 171);
moveTo(95, 162);
lineTo(84, 177);
moveTo(172, 183);
curveTo(180, 247, 187, 263);
curveTo(200, 294, 195, 334);
moveTo(171, 200);
curveTo(174, 230, 182, 258);
moveTo(171, 212);
curveTo(170, 226, 183, 274);
curveTo(196, 314, 190, 331);
moveTo(168, 219);
curveTo(168, 234, 176, 263);
moveTo(165, 224);
lineTo(168, 242);
moveTo(163, 227);
curveTo(161, 251, 184, 296);
moveTo(166, 240);
curveTo(170, 253, 171, 268);
moveTo(173, 261);
curveTo(175, 272, 180, 281);
curveTo(188, 295, 189, 330);
lineStyle(2, 0x000000, 100);
moveTo(68, 165);
lineTo(75, 154);
curveTo(68, 155, 62, 150);
curveTo(60, 150, 55, 154);
lineTo(50, 155);
lineTo(46, 163);
curveTo(49, 163, 50, 161);
curveTo(55, 162, 57, 166);
curveTo(62, 162, 68, 165);
moveTo(47, 159);
lineTo(58, 159);
curveTo(58, 156, 55, 157);
```

```
moveTo(44, 154);
lineTo(39, 148);
lineTo(18, 153);
curveTo(20, 160, 14, 167);
lineTo(30, 161);
curveTo(38, 164, 43, 163);
curveTo(40, 156, 44, 154);
moveTo(42, 157);
lineTo(36, 159);
lineTo(30, 157);
moveTo(44, 154);
curveTo(50, 150, 52, 153);
moveTo(44, 164);
curveTo(38, 167, 32, 166);
lineTo(22, 170);
lineTo(14, 170);
lineTo(4, 177);
curveTo(15, 179, 13, 191);
curveTo(22, 173, 43, 166);
curveTo(53, 182, 41, 203);
curveTo(53, 195, 62, 209);
lineTo(61, 196);
moveTo(57, 168);
curveTo(50, 169, 48, 163);
lineStyle(1, 0x000000, 100);
moveTo(31, 171);
curveTo(13, 201, 21, 210);
moveTo(37, 168);
curveTo(15, 204, 24, 214);
moveTo(31, 187);
curveTo(20, 213, 31, 218);
moveTo(35, 195);
curveTo(27, 225, 37, 224);
moveTo(34, 206);
curveTo(35, 224, 40, 222);
moveTo(44, 167);
curveTo(33, 221, 44, 225);
moveTo(41, 209);
curveTo(42, 228, 52, 228);
moveTo(47, 199);
curveTo(43, 228, 57, 227);
moveTo(49, 199);
curveTo(50, 226, 60, 228);
curveTo(71, 227, 66, 202);
moveTo(64, 200);
curveTo(62, 244, 54, 201);
curveTo(61, 240, 60, 204);
lineStyle(2, 0x000000, 100);
```

```
moveTo(189, 169);
curveTo(192, 166, 195, 169);
curveTo(199, 174, 206, 176);
curveTo(196, 181, 196, 191);
curveTo(190, 178, 181, 179);
curveTo(188, 182, 188, 186);
curveTo(189, 189, 194, 188);
moveTo(187, 173);
lineTo(193, 175);
moveTo(197, 184);
lineTo(208, 189);
lineTo(204, 193);
curveTo(199, 190, 189, 205);
curveTo(188, 192, 184, 190);
curveTo(182, 189, 182, 183);
curveTo(179, 182, 180, 179);
lineStyle(1, 0x000000, 100);
moveTo(198, 193);
curveTo(207, 207, 193, 216);
moveTo(194, 203);
curveTo(195, 215, 188, 221);
moveTo(189, 204);
curveTo(192, 214, 186, 225);
curveTo(183, 227, 178, 224);
moveTo(180, 181);
curveTo(178, 186, 181, 191);
curveTo(187, 210, 182, 224);
moveTo(177, 181);
curveTo(176, 189, 180, 198);
curveTo(186, 213, 179, 223);
moveTo(181, 202);
curveTo(181, 219, 177, 218);
moveTo(176, 182);
curveTo(178, 199, 174, 205);
lineStyle(2, 0x000000, 100);
moveTo(80, 248);
curveTo(74, 248, 79, 253);
lineTo(79, 259);
lineTo(85, 265);
moveTo(103, 293);
curveTo(113, 319, 123, 335);
moveTo(76, 249);
curveTo(76, 259, 70, 263);
curveTo(84, 271, 90, 283);
moveTo(106, 318);
lineTo(112, 336);
moveTo(69, 263);
curveTo(29, 284, 29, 295);
```

```
curveTo(18, 331, 16, 372);
curveTo(33, 377, 55, 374);
curveTo(56, 369, 68, 369);
curveTo(73, 375, 65, 377);
moveTo(47, 292);
curveTo(56, 312, 69, 329);
curveTo(63, 334, 63, 340);
moveTo(57, 312);
curveTo(60, 339, 69, 356);
lineTo(67, 358);
moveTo(65, 353);
lineTo(69, 370);
lineStyle(1, 0x000000, 100);
moveTo(18, 363);
curveTo(40, 367, 48, 365);
curveTo(55, 359, 66, 360);
moveTo(19, 374);
curveTo(18, 387, 20, 400);
moveTo(64, 370);
curveTo(65, 388, 67, 400);
moveTo(83, 255);
curveTo(83, 260, 80, 261);
moveTo(137, 248);
curveTo(137, 259, 139, 269);
moveTo(138, 263);
curveTo(142, 267, 148, 268);
lineStyle(2, 0x000000, 100);
moveTo(138, 254);
curveTo(156, 263, 157, 310);
curveTo(161, 327, 162, 335);
moveTo(141, 258);
curveTo(156, 270, 173, 273);
moveTo(159, 269);
curveTo(168, 289, 166, 314);
curveTo(167, 328, 169, 336);
moveTo(193, 287);
lineTo(202, 298);
curveTo(208, 336, 222, 365);
lineTo(215, 373);
lineTo(213, 374);
lineTo(195, 371);
moveTo(186, 307);
curveTo(181, 333, 185, 337);
lineTo(186, 342);
moveTo(63, 344);
curveTo(72, 338, 84, 337);
curveTo(103, 338, 140, 335);
curveTo(170, 335, 190, 347);
```

```
lineTo(187, 353);
moveTo(183, 346);
curveTo(205, 368, 185, 396);
lineStyle(1, 0x000000, 100);
moveTo(213, 373);
lineTo(213, 400);
moveTo(136, 329);
curveTo(136, 331, 139, 335);
lineStyle(2, 0x000000, 20);
moveTo(131, 324);
curveTo(132, 325, 136, 329);
moveTo(139, 324);
curveTo(138, 325, 136, 329);
lineStyle(4, 0x000000, 10);
moveTo(131, 284);
curveTo(141, 280, 151, 282);
moveTo(119, 284);
curveTo(109, 280, 100, 284);
moveTo(123, 280);
curveTo(126, 282, 128, 279);
}
```

رسم طائر

```
_root.createEmptyMovieClip("m",1);
        with(_root.m){
            beginGradientFill(fillType,colorss,alphas,ratios,matrix,spreadMethod,i
nterpolationMethod,focalPointRatioo);
                        lineStyle(10,0x0000ff,30,"none","round","round",1.11);
                        moveTo(600,0);
                        lineTo(0,0);
                        lineTo(0,200);
                        lineTo(600,200);
                        lineTo(600,0);
                        endFill();
        }

_root.createEmptyMovieClip("moosa",3);

       with(_root.moosa){
    fillType=["radial"]
    colors=[0xff0000,0x00ccff,0xccff00];
    colorss=[0xfff00f0,0x0fccff,0xcfff00];
    alphas=[100,100,100];

    ratios=[0,180,255];
    ratioss=[0,100,200];
    matrix={a:100, b:-
50, c:0, d:100, e:100, f:0, g:300, h:100, i:1};// matrixType:"box" , x:260 ,y:200 ,
```

```
w:50 , h:50 , r:(45/180)*Math.PI
   spreadMethod="reflect";              //  repeat , pad , reflect
أسلوب الإمتداد أو التكرار
   interpolationMethod="RGB";      // linearRGB , RGB    جودة الألوان
   focalPointRatio=  0.9 ;//      number       نقطة الإنحراف البؤري
   focalPointRatioo=  -0.9 ;
   lineStyle(9);
   lineGradientStyle(fillType,colors,alphas,ratioss,matrix,spreadMethod,interp
olationMethod,focalPointRatio);
   beginGradientFill(fillType,colors,alphas,ratios,matrix,spreadMethod,interpol
ationMethod,focalPointRatio);
                              lineStyle(8,0xfff00f,70); // الجسم
                              moveTo(426,100);
                              curveTo(270,240,150,90);
                              curveTo(120,50,100,90);
                              curveTo(80,65,100,40);
                              curveTo(120,15,160,65);
                              curveTo(300,10,426,100);
                               endFill();
                              lineStyle(3,0xffff33,50);  // المنقار
                              beginFill(0xffcc99,100);
                              moveTo(550,60);
                              curveTo(546,30,450,40);
                              curveTo(430,75,550,60);
                              endFill();
                              lineStyle(10,0xfff000,30);  // خط المنقار
                              moveTo(450,40);
                              curveTo(480,65,546,55);
                              moveTo(452,41);
                              lineStyle(6,0xfff00f,50);
                               beginFill(0xffccff,80);   // الرآس
                              curveTo(410,-30,380,50);
                              curveTo(401,90,350,150);
                              curveTo(450,110,452,41);
                              endFill();
                              lineStyle(2,0x000000,100);
                              beginFill(0x3333ff,100); // العين
                              moveTo(425,30);
                              curveTo(415,35,425,40);
                              curveTo( 435,35,425,30);
                              endFill();
                              lineStyle(6,0x00cc00,30);
                              moveTo(425,30);    // هدب العين
                              curveTo(440,35,425,45);
                              curveTo(435,35,425,30);
                              moveTo(450,41);     // خط الرأس
                              curveTo(440,100,381,50);
                              lineStyle(6,0x449944,30);
```

```
                    curveTo(420,130,450,41);
                    lineStyle(6,0x440044,30);
                    moveTo(450,41);        //الهدب
                    curveTo(420,-10,375,50);
                     moveTo(426,100);    lineStyle(9,0xffffff,30);     //
الجناح
                     curveTo(350,-10,100,80);  lineStyle(15,0xffffff,30);
                     curveTo(350,10,426,100); lineStyle(9,0xffffff,30);
                     curveTo(350,30,100,150); lineStyle(9,0xffffff,30);
                     curveTo(350,50,426,100);
          }
```

الرسم بالاكشن اسكريبت ٣,٠

رسم دائرة و تلوينها باللون البرتقالى :

```
import flash.display.*;
var circle:Shape = new Shape()
var xPos:Number = 100;
var yPos:Number = 100;
var radius:Number = 50;
circle.graphics.beginFill(0xFF8800);
circle.graphics.drawCircle(xPos, yPos, radius);
this.addChild(circle);
```

رسم دائرة و كتابة كلمة hello فى منتصفها

```
var mySprite:Sprite = new Sprite();
mySprite.graphics.beginFill(0xFFCC00);
mySprite.graphics.drawCircle(30, 30, 30);
var label:TextField = new TextField();
label.text = "hello";
label.x = 20;
label.y = 20;
mySprite.addChild(label);
this.addChild(mySprite);
```

استخدام الماوس فى رسم خطوط كما تشأ

```
package {
    import flash.display.Sprite;
    import flash.events.MouseEvent;

    public class Michael1 extends Sprite {
        private var _sprite:Sprite;

        public function Michael1(  ) {
            _sprite = new Sprite(  );
            addChild(_sprite);
            _sprite.graphics.beginFill(0xffffff);
            _sprite.graphics.drawRect(0, 0, 400, 400);
            _sprite.graphics.endFill(  );
```

```
_sprite.addEventListener(MouseEvent.MOUSE_DOWN, onMouseDown);
            _sprite.addEventListener(MouseEvent.MOUSE_UP, onMouseUp);
        } private function onMouseDown(event:MouseEvent):void {
            _sprite.graphics.lineStyle(1, 0, 1);
            _sprite.graphics.moveTo(mouseX, mouseY);
            _sprite.addEventListener(MouseEvent.MOUSE_MOVE, onMouseMove);
        }

        private function onMouseUp(event:MouseEvent):void
        {
            _sprite.removeEventListener(MouseEvent.MOUSE_MOVE, onMouseMove);
                                                                                             }
  private function onMouseMove(event:MouseEvent):void {
            _sprite.graphics.lineTo(mouseX, mouseY);
        }
    }
                                                                                             }
```



الــــفـــــصـــــل الـــثـــامـــن

استخدام مجيبات الأحداث Using event handler methods

مجيب الحدث Event Handler هو إجراءٌ فرعيٌّ يتمّ استدعاؤه تلقائيًّا في كل مرة يحدث فيها الحدث.. فمثلاً في حالتنا هذه، الحدثُ هو ضغط الزر، لهذا في كل مرة يضغط فيها المستخدم هذا الزر، يتم استدعاء الإجراء onPress وما عليك الآن سوى كتابة الأوامر في هذا الإجراء، لتتفاعل مع حدث ضغط الزر.
ويتكون اسم الإجراء الذي يستجيب للحدث، من اسم الأداة واسم الحدث object.eventMethod .

الصيغة العامة:-

```
object.eventMethod = function () {
  // Your code here, responding to event.
}
```

مثال :

```
next_btn.onPress = function () {
  nextFrame();
}
```

صيغة اخرى :-

```
// Assign a function reference to button's onPress event handler.
next_btn.onPress = goNextFrame;
// Define goNextFrame() function.
function goNextFrame() {
  nextFrame();
}
```

لاحظ التالى!!

```
// خطا
next_btn.onPress = goNextFrame();
// صح
next_btn.onPress = goNextFrame;
```

انشـــاء مربع نــص بـــالــكـــود

لأنشاء مربع نص و تحديد نوعه بحيث يكون لادخال النصوص نستخدم الكود التالى :

```
this.createTextField("my_txt", 99, 10, 10, 200, 20);
my_txt.border = true;
my_txt.type = "input";
this.createTextField("myOther_txt", 100, 10, 50, 200, 20);
myOther_txt.border = true;
myOther_txt.type = "input";
myOther_txt.onSetFocus = function(my_txt:TextField) {
  my_txt.text = "I just lost keyboard focus";
        }
```

تحميل صورة داخل الفلاش بالاكشن اسكريبت

نستخدم الكود التالى :-

```
loadMovie("myBitmap.jpg", "myClip_mc");
```

فى اكشن اسكريبت ٣ نستخدم الكود التالى

```
package {
  import flash.display.*;
  import flash.net.URLRequest;
  public class LoaderExample extends Sprite {
    public function LoaderExample(  ) {
      // 1. نقوم بعمل كائن للتعامل مع الكلاس
      var loader:Loader = new Loader(  );
      addChild( loader );
      loader.load( new URLRequest( "image.jpg" ) );
    }
  }
}
```

الــــفــــصـــــل الـــتـــاســــع

ا لتحكم فى الصوت داخل الفيلم باستخدام الاكشن

نقوم بعمل كائن جديد للصوت بالطريقة التالية :-

```
var soundObjectName:Sound = new Sound (Target);
```

مثال :

اذا كان لدينا movieclip اسمه myMovieClip_mc و نريد التحكم فى الصوت الموجود بداخله نستخدم الكود التالى :

```
var mySound:Sound= new Sound ("myMovieClip_mc");
```

ثم بعد ذلك نتحكم فى مستوى الصوت من خلال الكود التالى :-

```
mySound.setVolume (50);
```

لتحميل ملف mp3 من خارج ملف الفلاش يجب عمل كائن للصوت

```
var myFavMP3:Sound = new Sound();
```

ثم تحميل الملف الصوت mySong.mp3

```
myFavMP3.loadSound("mySong.mp3", true);
```

لاحظ اننا استخدمنا المعطى true لكى يقوم ملف الصوت بالعمل بعد تحميل swf و لكن اذا كتبنا false فيصبح الكود كالتالى :

```
myFavMP3.loadSound("mySong.mp3", false);
```

لكن يجب ان نستخدم دالة start لكى يبدأ الصوت فى العمل

```
myFavMP3.start();
```

ଓଃଓଃଓଃଓଃ

الـــتـــحـــكـــم فـــى الـــفـــيـــلـــم

احيانا نحتاج لوقف الفيلم عند فريم معين فنستخدم الدالة

```
stop ();
```

اما للتنقل بين الفريم و الذهاب من فريم الى اخر نستخدم الامر التالى

```
gotoAndPlay(7);
gotoAndPlay(20);
// ممكن ان نستخدم اسم الفريم بدلا من رقمه
gotoAndPlay("my frame label")
```

اما للذهاب الى Scene اخرى

```
gotoAndPlay("My Scene","My Frame");
```

ଓଃଓଃଓଃଓଃ

Moving and Changing Movie Clips تحريك و تعديل الـ

كل عنصر فى ساحة العمل له موقع محدد يقاس بالبكسيل pixels حيث ان الجزء الايسر العلوى من مسرح العمل يعتبر ٠,٠ و هى اول نقطة للألتقاء المحور الافقى و الرأسى لمسرح العمل .
و لكى نعرف مكان عنصر على مسرح العمل نستخدم الخاصيتين _x and _y

مثال لتحديد مكان movieclip :-

نقوم بعمل movieclip و نقوم بتسميته myclip1 ثم نكتب فى خانة instace الاسم الذى نختاره للتعامل مع هذا الــmovieclip و ليكن اسمه myClip ثم نكتب الكود التالى :

```
trace(myClip._x);
trace(myClip._y);
```

فيظهر الناتج التالى مثلا و هذا يتوقف على مكان الـmovieclip

```
268
115.5
```

و يمكنك تغيير مكانه من خلال الكود التالى :-

```
myClip._x = 200;
myClip._y = 250;
```

مــعرفة مكان الفأرة فى الفيلم Mouse Location

لكى نعرف موقع الفأرة على مسرح العمل نستخدم الخاصيتين_xmouse و _ymouse
مثال : قم بعمل Movie clip و قم بتحديده و اضغط f9 و اكتب الكود التالى :-

```
onClipEvent (enterFrame) {
   trace(_root._xmouse);
   trace(_root._ymouse);
   trace("");
}
```

فتلاحظ فى شاشة المخرجات انك كلما تحركت بالماوس تغيرت الاحداثيات لاننا استخدمنا الحدث enterFrame .
كود أخر لكن يجب عمل ٢ مربع نص و اسم الاول box1_txt و الثانى box2_txt و نضع الكود التالى فى اول فريم فى الفيلم :-

```
var mouseListener:Object = new Object();
mouseListener.onMouseMove = function() {
  // تقوم الدالة بأرجاع الاحداثى السينى و الصادى للماوس
// return X and y
  box1_txt.text = _xmouse;
  box2_txt.text = _ymouse;
};
Mouse.addListener(mouseListener);
```

❈ ❈

عـمـل تـدويـر لـلـمـوفـى كـلـيـب Movei Clip Rotation

مثال : قم بعمل موفى كليب واعطيه الاسم myClip واضغط على الفريم رقم ١ و اضغط f9 و اكتب الكود التالى :

```
myClip._rotation = 90;
myClip._rotation++;
_root["myClip"]._rotation = 45;
this._rotation += 0.5;
```

★★★★★★★★★★★★★★★★★★★

Stretching and Shrinking Movie Clips عمل تمدد للموفى كليب

***_xscale باستخدام الخاصية ***

مثال قم بعمل موفى كليب و سوف نشير اليه فى الكود من خلال التعبير this .

```
onClipEvent (load) {
   // نعرف الاحداثيات الاصلية للموفى كليب
   origWidth = this._width;
   origHeight = this._height;
}
onClipEvent (enterFrame) {
//معرفة المسافة بين منتصف الموفى كليب و الفأرة
   dx = _root._xmouse-this._x;
   dy = _root._ymouse-this._y;
//حساب النسبة المئوية للتمدد
   sx = 100*dx/(origWidth/2);
   sy = 100*dy/(origHeight/2);
```

```
// تحديد قيمة التمدد و اسنادها للموفى كليب
this._xscale = sx;
this._yscale = sy;
}
```

☜ يمكننا عمل محاكاة لحركة الـ3D بالتمدد للموفى كليب من خلال الكود التالى :-

```
onClipEvent(load) {
    scaleAmt = 10;
    x = 525;
    y = 25;
}
onClipEvent(enterFrame) {
    scaleAmt++;
    x -= 5;
    y += 5;
    this._xscale = scaleAmt;
    this._yscale = scaleAmt;
    this._x = x;
    this._y = y;
}
```

** كما يمكن كتابة نفس الكود بطريقة اخرى ونختصر بعض السطور كما يلى :-

```
onClipEvent(load) {
    this._x = 525;
    this._y = 25;
}
onClipEvent(enterFrame) {
    this._xscale++;
    this._yscale++;
    this._x -= 5;
    this._y += 5;
}
```

꧁꧂꧁꧂꧁꧂꧁꧂꧁꧂꧁꧂

التحكم فى رؤية العناصر الموجودة على مسرح العمل فى برنامج فلاش Visibility

```
myClip._visible = false;
```

الخاصية visible تأخذ معاملين اما true او false اى يظهر او لا .

```
myClip._alpha = 50;
```

الخاصية alpha شدة الوضوح تأخذ قيم من ٠ حتى ١٠٠
لعمل تأثير الـFading

```
onClipEvent(enterFrame) {
  if (this._alpha > 0) {
      this._alpha -= 5;
    }
```

```
}
```

Selecting and Dragging Movie Clips السحب و الالقاء للموفى كليب

ترتبط الخاصية hitTest بعملية السحب و الالقاء حيث ان هذه الخاصية هى المسئولة عن معرفة هل الفأرة قامت بلمس الموفى كليب على مسرح العمل ام لا
و لكن ذلك مع استخدام الحدث المناسب و هو onClipEvent(mouseDown) or onClipEvent(mouseUp) و يمكنك تجربة الكود التالى :

```
onClipEvent (mouseUp) {
   this.gotoAndStop(2);
}
```

و لكن بفرض ان هناك اكثر من موفى كليب على مسرح العمل فيمكنك معرفة اى منهم تم النقر علية بالفأرة من خلال الكود التالى :

```
onClipEvent (mouseUp) {
   if (this.hitTest(_root._xmouse, _root._ymouse)) {
      this.gotoAndStop(2);
   }
}
```

السحب و الافلات Dragging :-

```
onClipEvent (mouseDown) {
   if (this.hitTest(_root._xmouse, _root._ymouse)) {
      this.startDrag();
   }
}
onClipEvent (mouseUp) {
   if (this.hitTest(_root._xmouse, _root._ymouse)) {
      this.stopDrag();
   }
}
```

❁❁❁❁❁❁❁❁❁❁❁❁❁❁❁❁❁❁❁❁❁❁❁❁

الفصل الحادى عشر دوال الرياضيات

****الكائن :- math** قد قمنا بتجربة هذا الكائن من قبل فهو يتيح إجراء العمليات الحسابية والتعامل مع الأرقام في لغة الأكشن . ولهذا الكائن العديد من الأساليب (**methods**) والخصائص التي تساعد في تنويع وإنجاز العمليات الحسابية..

ويضم هذا الجدول تفصيل لوظائف بعض الأساليب والخصائص:-

الأسلوب	الوصف
() Math.abs	وهو يعود بقيمة القيمة المطلقة للعدد الموجود بين القوسين...
() Math.sin	وهو يعود بقيمة الـ cos للعدد الموجود بين القوسين..
()Math.cos	وهو يعود بقيمة الـ sin للعدد الموجود بين القوسين..
() Math.tan	وهو يعود بقيمة الـ tan للعدد الموجود بين القوسين..
() Math. pow	ويتم كتابة عددين داخل القوسين (x,y) بحيث يعود الكائن بنتيجة رفع قيمة ال xكقوى
() Math.sqrt	وهو يعود بقيمة الجذر التربيعي للعدد بين القوسين..
()Math.round	وهو يقوم بتقريب القيمة الموجودة بين القوسين لأقرب رقم صحيح.....
()Math.max	وهو يقارن العددين المكتوبة بين القوسين "طبعا يفصل بينهما فاصلة" ويعود بالرقم الأك ...
() Math.min	وهو يقارن العددين المكتوبة بين القوسين "طبعا يفصل بينهما فاصلة" ويعود بالرقم الأص ...
() Math.log	وهو يعود بلوغاريتم العدد بين القوسين على أن يكون أكبر من الصفر....
()Math.PI	والذي يعطي باختصار قيمة الثابت PI وهو ٣,١٤ ... وهو الوحيد من الأوامر السابقة الذ يحتاج إلى وسائط بين القوسين..
( )Math. ceil	وهو يقوم بتقريب القيمة بين القوسين أقرب أكبر عدد صحيح..
( )Math.floor	وهو يقوم بتقريب القيمة بين القوسين أقرب أقل عدد صحيح

بافتراض أن يكون لديك مربعين نص من النوع input text لهما الأسماء num1_txt و num2_txt

ومربع نص من النوع dynamic text اسمه result_txt لعرض النتيجة..

و٧ أزارير على الشاشة ٦ منها تؤدي عمليات حسابية على حسب أسمائها ، والأخير لمسح جميع مربعات النص

الزر math.abs اكتب له الأوامر التالية:

```
1 on(release)
2 { result_txt.text=Math.abs(num1_txt.text);}
```

والزر math.round :

```
1 on(release)
2 { result_txt.text=Math.round(num1_txt.text);}
```

والزر math.min لمقارنة عددين:

والزر math.log:

```
1 on(release)
2 { result_txt.text=Math.log(num1_txt.text);}
```

والزر math.pow:

```
1 on(release)
2 { result_txt.text=Math.pow(num1_txt.text,num2_txt.text);}
```

والزر math.sqrt :

كالتالي أما الزر المستخدم للمسح:

```
1 on(release)
2 {   num1_txt.text=" ";
3     num2_txt.text=" ";
4     result_txt.text=" ";}
```

الفصل الثانى عشر

اصطياد المفاتيح Detecting Keypresses

معرفة المفتاح الذى تم الضغط علية فى لوحة المفاتيح من خلال اسناد الكود التالى الى موفى كليب :

```
on (keyPress "a") {
   trace("Key 'a' pressed.");
}
```

و هذة قائمة بالمفاتيح التى يمكن تجربة الكود السابق معها

<Left>	<End>	<PageUp>
<Right>	<Insert>	<PageDown>
<Up>	<Delete>	<Tab>
<Down>	<Backspace>	<Escape>
<Home>	<Enter>	<Space>

مثال أخر

```
on (keyPress "<Left>") {
   trace("Left pressed.");
}
```

أما اذا كنت تريد معرفة اذا كان المفتاح يتم الضغط علية حاليا و الضغط مستمر مثلما فى الالعاب عندما يضغط اللاعب على مفتاح للأمام لكى تستمر الحركة فى اللعبة مثال :

```
if (Key.isDown(Key.LEFT)) {
```

```
    trace("The left arrow is down");
}
```

الدالة Key.isDown الناتج الخاص بها هو true او false و فى حالة الضغط المستمر يكون true .

قائمة بالمفاتيح الاخرى التى يمكن تجربة الكود السابق معها :

Key.BACKSPACE	Key.ENTER	Key.PGDN
Key.CAPSLOCK	Key.ESCAPE	Key.RIGHT
Key.CONTROL	Key.HOME	Key.SHIFT
Key.DELETEKEY	Key.INSERT	Key.SPACE
Key.DOWN	Key.LEFT	Key.TAB
Key.END	Key.PGUP	Key.UP

عمل متصنت لحركة المفاتيح Key Listeners

بداية نقوم بأخبار الفلاش اننا قمنا بعمل متصنت لحركة المفاتيح

```
Key.addListener(_root);
```

ثم بعد ذلك نقوم بعمل دالة لتتبع حركة المفتاح عندما يرفع المستخدم اصبعة عن المفتاح لكى نعرف الكود المقابل للمفتاح فى الاسكى كود مثال

```
_root.onKeyUp = function() {
    trace(Key.getAscii());
}
```

اضغط على الحرف a و كود الاسكى المقابل لة هو ٩٧

و باستخدام الكود التالى يمكنك تحويل كود الاسكى للحرف المقابل لة فى لوحة المفاتيح

```
trace(String.fromCharCode(Key.getAscii()));
```

مثال : افترض انك تريد عمل لعبة عبارة عن سيارة صغيرة تسير من خلال مفاتيح الاتجاهات فى لوحة المفاتيح فما هو الكود الازم لعمل ذلك .

فكر قليلا يمكنك استخدام ما تعلمناة فى الدروس السابقة لعمل هذة اللعبة بسهولة

ارى انك اقتربت من الحل و لكنك بحاجة لبعض المساعدة لذلك سوف نفكر سويا

بداية نحن نحتاج لعمل موفى كليب السيارة و لنفترض اننا قمنا برسم مربع صغير ثم قمنا بالضغط على f8 لتحويلة الى موفى كليب ثم قمنا بالنقر على الموفى كليب لتحديدة و انقر على f9 لتظهر شاشة محرر الكود ثم اكتب التالى

فى حدث تحميل الموفى كليب نريد ان نعرف احداثيات الموفى كليب و عرفنا متغير speed و اعطيناة القيمة ٥//

```
onClipEvent(load) {
    x = this._x;
    y = this._y;
    speed = 5;
}
onClipEvent(enterFrame) {

    if (Key.isDown(Key.LEFT)) {
        x -= speed;
    }
        if (Key.isDown(Key.RIGHT)) {
        x += speed;
    }
        if (Key.isDown(Key.UP)) {
        y -= speed;
    }
        if (Key.isDown(Key.DOWN)) {
```

```
            y += speed;
        }

        this._x = x;
        this._y = y;
    }
```

و يمكن كتابة الكود السابق بطريقة اخرى ليعطى نفس النتيجة مع كتابة الاتجاة على مسرح العمل :
فى البداية قم بعمل موفى كليب السيارة و قم بتسميتة فى مربع : instance of
car_mc ثم قف على الفريم الاول للفيلم ثم اضغط f9 و اكتب فى نافذة الكود التالى :

```
// متغير رقمى لحساب المسافة
var distance:Number = 10;
```

انشأ مربع نص اثناء تنفيذ الكودو لاحظ انك تستخدم دالة انشاء مربع النص التى لها معطيات يجب ان تاخذها منها اسم مربع النص و العمق و مكانة x و الy والطول و العرض

```
this.createTextField("display_txt", 999, 0, 0, 100, 20);

var keyListener:Object = new Object();

keyListener.onKeyDown = function() {

  if (Key.isDown(Key.LEFT)) {
// لاحظ أستخدامنا للدالة Math و هى المسئولة عن العمليات الرياضية
//  و من خصائص هذة الدالة الخاصية max التى تعود بالقيمة الاكبر
    car_mc._x = Math.max(car_mc._x - distance, 0);

    display_txt.text = "Left";

  } else if (Key.isDown(Key.RIGHT)) {


//  و من خصائص هذة الدالة الخاصية min التى تعود بالقيمة الاصغر

    car_mc._x = Math.min(car_mc._x + distance, Stage.width -

  car_mc._width);

    display_txt.text = "Right";

  } else if (Key.isDown(Key.UP)) {

    car_mc._y = Math.max(car_mc._y - distance, 0);

    display_txt.text = "Up";

  } else if (Key.isDown(Key.DOWN)) {

    car_mc._y = Math.min(car_mc._y + distance, Stage.height -
```

```
car_mc._height);

  display_txt.text = "Down";

 }

};

Key.addListener(keyListener);
```

فى اكشن اسكربت ٣

```
package {
    import flash.display.Sprite;
    import flash.events.KeyboardEvent;

    public class michael1 extends Sprite {
        public function michael1(  ) {
            stage.focus = this;
            addEventListener(KeyboardEvent.KEY_DOWN,
onKeyDown);
        }

        private function onKeyDown(event:KeyboardEvent):void {
            trace("key down: " + event.charCode);
        }
    }
}
```

الفصل الثالث عشر

الوقت و التاريخDates and Times

معرفة الوقت و التاريخ الحالى

```
// Create a new Date object.
today = new Date(  );
// Displays client-side date and time
trace(today);
```

الناتج

Tue Jul 11 17:17:48 GMT+0300 2006

معرفة السنةو الشهر و اليوم و الساعة و الثانية

```
var dDateObject:Date = new Date(year, month, date, hour,
minute, second, millisecond);
```

معرفة السنة من تاريخ معين

```
var dWhen:Date = new Date(1978, 9, 13);
trace(dWhen.getYear()); // Displays: 78
trace(dWhen.getFullYear()); // Displays: 1978
```

مثال اخر قم بعمل مربع نص من الن dyna اسمه myMsg _txt نستخدمه لعرض الرسالة وزر اكتب فيه الأوامر التالية-

!

```
1 on(release)
2 { myDate=new Date();
3   myMsg_txt.text= "can u believe it's already the year "+ myDate.getFullYear()  + "?";}
```

ففي السطر ٢ قمنا بإنشاء كائن تاريخ جديد وسمينا هذا الكائن بـ myDate وبما أننا لم ندخل أي وسائط بين القوسين فإننا نعني استخدام الوقت والتاريخ الحاليين في الكمبيوتر..

وفي السطر ٣ حددنا الرسالة التي نريد ظهورها في مربع النص. myMsg ثم كتبنا اسم الكائن myDateمتبوعا بالـ methodالتي تسمى getFullYear () والتي ستعرض رقم العام الحالي ومن ثم سيتم عرضه ضمن الرسالة..

***طيب لو كتبنا بدل الأوامر السابقة الأوامر التالية ستعرض لي بعض ضغط المستخدم للزر في مربع النص myMsg الوقت الحالي بالساعة والدقيقة والثانية ...

!Error

```
on(release)
{ myDate=new Date();
  myMsg_txt.text= myDate.getHours()+" : "+ myDate.getMinutes()+ " : " +myDate.getSeconds();}
```

****وإذا أردت عمل ساعة لفيلمك فالأوامر مشابهة للسابق مع خطوة أخرى جديدة كالتالي:-

قم بعمل رمز من النوع movieclip وعندما تنتقل لنافذة القصاصة قم بإدراج طبقة ثانية..

الطبقة الأولى اضغط على الإطار الأول منها وقم بإنشاء مربع نص على النافذة من النوع dynamic textوسمّه time لتعرض الوقت فيه..

الطبقة الثانية عند الإطار الأول منها اكتب الأوامر التالية:-

!Error

```
1 myDate=new Date();
2 time_txt.text= myDate.getHours()+" : "+ myDate.getMinutes()+ " : " +myDate.getSeconds();
```

وعند الإطار ٢ منها اضغط على f6 ثم اكتب الأمر التالي والذي يسبب الانتقال للإطار ١ و تحديث الوقت بشكل دوري ومستمر--:

```
1 gotoAndPlay(1);
```

بعدها عد للنافذة الرئيسية واسحب الرمز من المكتبة وضعه على النافذة أمامك لتتكون لديك ساعة تعمل طوال الفيلم...

وسأستعرض الآن الأساليب (methods) المختلفة الخاصة بكائن الوقت:--

الـmethod	وصفه
getDate ()	يعود لنا برقم اليوم من الشـهر وهي قيمة عشرية تتراوح بين ١ إلى ٣١ مثل25
getDay ()	يعود لنا برقم اليوم من الأسبوع وهي قيمة عشرية من ٠ إلى ٦ فكل يوم من الأس برقم ابتداء من الصفر.
getFullyear ()	تعود برقم السنة الحالية وتكون مكونة من أربع قيم..مثل ٢٠٠٥
getHours ()	تعود بقيمة الساعة الحالية وهي تتراوح من ٠ إلى ٢٣
() getMilliseconds	وهي تعود بالأجزاء من الثانية وتتراوح بين ٠ إلى ٩٩٩
getMinutes ()	تعود بالدقائق وهي قيمة عشرية من ٠ إلى ٥٩..
getMonth ()	تعود بقيمة عشرية توضح لي الشهر الحالي طبعا الشهور الميلادية وليس الهجرية يناير له القيمة ٠ والشهر فبراير له القيمة ١ وهكذا إلى آخر شهر له القيمة..11
getSeconds ()	تعود لي بالثواني وهي قيمة عشرية تتراوح بين ٠ و59

وهناك العديد والعديد من الأساليب methods الخاصة بالكائن data تستطيع الوصول إليها عن طريق التسلسل التالي:--

علامة ال+ في لوحة الأكشن سكريبت >> Built-in Classes >> ثم Core >> ثم Data >> ثم method و بعدها تصل لجميع الأساليب الخاصة بهذا الكائن .وتستطيع إيجاد شرح كافي عن أي أمر في الأكشن سكريبت ... عن طريق **ActionScript Dictionary** الموجودة ضمن قائمة.... **help**

الفصل الرابع عشر

Using Dataاستخدام البيانات و المعلومات

يمكننا تخزين البيانات فى ملف نصى و استدعائها داخل الفلاش مثال :
قم بعمل ملف نصى باستخدام المفكرة و عرف فية متغيرين و اكتب فية التالى

```
color=red&area=100
```

ثم قم بحفظ الملف باسم variables.txt
ثم افتح ملف فلاش جديد و قم بحفظة فى نفس المجلد الذى بة الملف النصى .
ثم اكتب الكود التالى فى اول فريم فى الملف

```
//نستخدم الدالةLoadVars لتحميل متغيرات اثناء عرض الفيلم
var lvSampleData:LoadVars = new LoadVars();
// ثم نقوم بعمل دالة لنعرف منها هل تم تحميل البيانات ام لا
lvSampleData.onLoad = function(bSuccess:Boolean):Void {
        trace("data loaded");
      };
lvSampleData.load('variables.txt');
```

و عند تجربة العمل تجد النتيجة اما true او false
و يفترض انها true اذا كنت و ضعت الملفين فى نفس المكان على الهارد ديسك .
أما اذا كان نريد معرفة قيمةالمتغيرين color و area فاننا نستخدم الكود التالى :

```
lvSampleData.onLoad = function(bSuccess:Boolean):Void {
        if(bSuccess) {
```

```
            trace('color = ' + this.color);
            trace('area = ' + this.area);
          }
        else {
            trace('There is an error with the data.');
          }
        };
lvSampleData.load('variables.txt');
```

color = red :   الناتج
area = 100

اما اذا كنت تريد عرض محتويات الملف النصى داخل الفلاش فيمكنك استخدام الكود التالى :

```
var lvExample:LoadVars = new LoadVars();
        lvExample.onData = function(sText:String):Void {
          trace(sText);
        };
        lvExample.load('variables.txt');
```

الناتج   color=red&area=100

مثال أخر

قم بعمل ملف نصى و قم بتسمية electricBill.txt و اكتب فية الكود التالى :

```
&electricBill=60
```

ثم قم بعمل ملف فلاش جديد و اكتب الكود التالى فى الفريم الاول

```
//انشأ مربع نص بالكود لكى نستقبل فية قيمة المتغير المكتوبة فى الملف النصى.
this.createTextField("my_txt", 99, 10, 10, 200, 20);
my_txt.border = true;
my_txt.type = "input";
//استخدمنا الدالة () Loadvars لتحميل المتغير
    var externalData:LoadVars = new LoadVars();

    externalData.onLoad = function(){
//عند تحميل الملف النصى و قراءة محتواة تتغير قيمة مربع النص my_txt لتصبح قيمة المتغير  electricBill.
      my_txt.text = externalData.electricBill;

    }
    externalData.load("Electric_Bill.txt");
```

الناتج:

مربع نص مكتوب بة ٦٠

<u>كيف يتعامل الفلاش مع ملفات التخزين الخارجية مثل لغة الXml</u>

ملحوظة مرفق فى اخر الكتاب ملحق عن لغة الXml بالتفصيل يجب الرجوع الية قبل قراءة هذا الفصل و بعد ذلك يمكنك متابعة القراءة و تنفيذ الاكواد :

استخدام the XML class

تستخدم فى تحميل و ارسال البيانات الى السيرفر من خلال HTTP POST method

- The load()تقوم بتحميل ملف اكس ام ال من رابط و تضعة داخل كود الاكشن اسكريبت

- **The send() تقوم بارسال البيانات و اعادة البيانات مرة اخرى**
- **The sendAndLoad() تقوم بارسال البيانات و تحميل الناتج**

The XML Objectالكائن

هذا الكائن يحتوى على مجموعة من الدوال و الخصائص التى تساعدك على التعامل مع لغة الxml لذلك يجب فى البداية عمل متغير من هذا الكائن لنستطيع التعامل معة .

```
myXML = new XML();
```

و يمكننا كتابة كود الxml داخل كود الاكشن اسكربت بواسطة الامر parseXML الذى يمكنك من كتابة الxml كأنها string

```
myXML = new XML();
myXML.parseXML("<user><name>Gary</name><ID>47</ID></user>");
```

طريقة اخرى مختصرة

```
myXML = new XML("<user><name>Gary</name><ID>47</ID></user>");
```

و لكن كيف نتأكد من اننا كتبنا الامر السابق بشكل صحيح دون ان نخطأ فى شئ بالنسبة لكود الxml بالطبع هناك دالة جاهزة لذلك و هى

```
trace(myXML.status);
```

اذا كان الناتج

-٩ معنى ذلك ان هناك تاج فتح و لم يقفل.

-١٠ معنى ذلك ان هناك تاج قفل و لم يكن مفتوح من قبل .

اما اذا كان الناتج0 فمعنى هذا ان الكود مكتوب بطريقة صحيحية .

استخراج البيانات من ملف xml

```
myXML = new XML("<user><name>Gary</name><ID>47</ID></user>");
trace(myXML.firstChild);
```

"<user><name>Gary</name><ID>47</ID></user>". الناتج

```
myXML = new XML("<user><name>Gary</name><ID>47</ID></user>");
trace(myXML.firstChild.firstChild);
```

<name>Gary</name> **الناتج**

<u>طريقة اخرى باستخدام childNodes</u>

```
myXML = new XML("<user><name>Gary</name><ID>47</ID></user>");
trace(myXML.childNodes[0].childNodes[0]);
```

```
myXML = new XML("<user><name>Gary</name><ID>47</ID></user>");
trace(myXML.childNodes[0].childNodes[0].childNodes[0]);
```

الناتج : Gary

```
myXML = new XML("<user><name>Gary</name><ID>47</ID></user>");
```

```
trace(myXML.childNodes[0].childNodes[1].childNodes[0].nodeValue);
```
النتاج : ٤٧

عمل ملف xml من البداية :

```
myXML = new XML();
newElement = myXML.createElement("user");
myXML.appendChild(newElement);
newElement = myXML.createElement("name");
myXML.childNodes[0].appendChild(newElement);
newText = myXML.createTextNode("Gary");
myXML.childNodes[0].childNodes[0].appendChild(newText);
newElement = myXML.createElement("ID");
myXML.childNodes[0].appendChild(newElement);
newText = myXML.createTextNode("47");
myXML.childNodes[0].childNodes[1].appendChild(newText);
```

If you want to change the value of a text node, you need to set the nodeValue property:

اذا كنت تريد تغيير قيمة النص فى عقدة يجب عليك اضاف خاصية nodevalue

```
myXML.childNodes[0].childNodes[1].childNodes[0].nodeValue = 53;
```

```
Attributes
<user>
   <name type="alias">Gary</name>
   <ID>47</ID>
</user>
```

مثال :

```
myXML = new XML("<user><name type='alias'>Gary</name><ID>47</user>");
trace(myXML.childNodes[0].childNodes[0].attributes.alias);
```
طريقة اخرى
```
trace(myXML.childNodes[0].childNodes[0].attributes["alias"]);
```
يمكنك معرفة عدد العقد فى ملف الxml

```
myXML = new XML("<user><name>Gary</name><ID>47</user>");
trace(myXML.childNodes[0].childNodes.length);
```
الناتج ٢

يمكنك معرفة اسم العقدة ايضا
```
myXML = new XML("<user><name>Gary</name><ID>47</user>");
trace(myXML.childNodes[0].childNodes[0].nodeName);
```

مثال على ملف xml

```
<book>
<title>ActionScript Arabic book</title>
<authors>Michael nabil</authors>
```

</book>

الان مثال على كيفية عرض البيانات من داخل ملف ال xml السابق
قم بحفظ ملف ال xml السابق باسم books.xml
ثم افتح ملف فلاش جديد و قم بحفظة فى نفس المجلد الذى بة الملف book.xml
ثم اكتب الكود التالى فى اول فريم فى الملف

```
// book_xml و اسمة xmlنقوم بعمل كائن جديد من الكلاس
var book_xml:XML = new XML();
//xml نجبر الفلاش على تجاهل المسافات البيضاء فى ملف
this.book_xml.ignoreWhite = true;

// xml نقوم بتحميل ملف ال
this.book_xml.load("book.xml");
```

جدول بة الخصائص والدوال التى سبق و استخدمناها فى الامثلة السابقة ارجو ان تقوم بتجربتها لتعرف ما هى فائدتها .

Properties, Methods, and Return Types of the XML Object

الخصائص و الدوال الخاصة بالكائن xml

PROPERTY AND METHODS	RETURN TYPES
Attributes	Returns an associative array containing all of the attributes of the specified node
ChildNodes	Read-only; returns an array containing references to the child nodes of the specified node
FirstChild	Read-only; references the first child in the list for the specified node
hasChildNodes()	Returns true if the specified node has child nodes; otherwise, returns false
LastChild	References the last child in the list for the specified node
nextSibling	Read-only; references the next sibling in the parent node's child list
NodeName	The node name of an XML object
NodeType	The type of the specified node (XML element –type = 1, text node –type = 3)
NodeValue	The text of the specified node if the node is a text node
ParentNode	Read-only; references the parent node of the specified node
previousSibling	Read-only; references the previous sibling in the parent node's child list
RemoveNode()	Removes the specified node from its parent
toString()	Converts the specified node and any children to XML text

الفصل الخامس عشر

البرمجة بالكائنات

Object-Oriented Programming

Object : وتعني شئ أو كائن (حاجة) ... حيت يعتبر الكرسي Object ، والسيارة أيضاً Object أو شئ ... وهكذا ...

Oriented: مثلاً لو أننا تعرفنا على شخص ما وكان همه الوحيد في هذه الدنيا المال ، فأننا نستطيع أن نطلق عليه أنه Money-oriented person أو شخص تحركه المادة ، حيث أن المال هو المحرك الأساسي في حياته.

Object-oriented Programming: والمقصود بها أن المبرمج يحلل العمل المراد منه وبناءه على أساس أنه مجموعة من الأشياء منفصلة Objects ، ولكنها في نفس الوقت تتعامل مع بعض البعض على أسس منظمة ومحددة. مثلاً في منظومة المكتبة يمكننا أن نعتبر أن الطالب Object ، وله أسم ورقم ، والعمليات التي يمكن أن يقوم بها في المكتبة هي القراءة والاستعارة وترجيع كتب... وهكذا...

OOP هي أولاً وأخيراً طريقة تصميم وبرمجة ، ولا تقدم أي ميزة اضافية على المنتج النهائي للبرمجيات (على سبيل المثال Execution Files او DLL Files) ، ولكنها في نفس الوقت للمبرمج الذي يستخدمها تقدم ميزات تسهل عليه التعامل مع البرامج التي هو يكتبها أو غيره من المبرمجين ، من حيث سهولة إدارة البرامج وإعادة استخدامها من قبل فرق العمل ، أيضاً لسهولة تخيل البرامج لأنها مصصمة من البداية على أسس OOP وتكمن سهولةOOP في محاكاتها للكائنات.

يولي البعض أهمية كبيرة لمفاهيم كبيرة وكثيرة للبرمجة الشيئية ، فهم من جهة لا يدركون أي أهمية لمصطلحاتها وتقنياتها ، ومن جهة أخرى فهم لا يستطيعون الفرار من قوقعة التفكير الهيكلي .... لذلك نجد أن كثيراً من البرامج التي كتبت والتي ما زالت حتى الآن مكتوبة بأسلوب هيكلي وليس بأسلوب شيئيي حتى وإن وجدت في الكود عبارات مثل: class و extends وغيرها ... تظل طريقة التفكير نفسها هي طريقة التفكير لدى تصميم أي

برنـامج آخـر بواسطـة الـدوال ، الان أحـاول ولـو بـشكل بـسيط معالجـة بعض المـسائل النظريـة المتعلقـة بالبرمجـة الشيئيـة دون أي تعمق في المسائل الكوديـة المختلفـة بين لغات البرمجـة.

+الفرق بين التفكير الهيكلي والتفكير الشيئي.
+التجريد.
+الكائنات والكبسلة.
+التماسك والتزاوج.
+الوراثـة وتعدد الأوجـه.
+أنماط التصميم البرمجيـة.

الفرق بين التفكير الهيكلي والتفكير الشيئي:

لحظـة تفكير المبرمج بالبرنـامج الجديد الذي يريد كتابتـه هي اللحظـة الفاصلـة التي تجعلـه يقرر بين إن كـان يفكر بشكل شيئي أو هيكلي ، إذا افترضنا أن البرنامج الجديد هو عبارة عن نظام لإدارة مدرسـة ابتدائيـة فإن غالبية قراء هذه المقالة سيبدؤون التفكير بتحديد مهـمـة هذا البرنـامج ، ثم بعد ذلك يتم تقسيم البرنـامج إلى مجموعـة كائنـات ، ويتم إسناد مهـمـة لكل كائـن .... طريقة هذه التفكير هي طريقـة التفكير الـهيكلي .. إن الـسؤال الأساسـي المفترض عند بداية تصميم أي برنامج ليس ماهي مهـمـة هذا البرنـامج ... بل مـا هـو العالم الـذي تريد منـي نمذجتـه فـي هذا البرنـامج ... وبالتالي فإن الإجابة ستكون نمذجـة عالم المدرسـة الابتدائيـة ... إذا رجعـنا لطريقة التفكير الـهيكلـي فإن الإجابة ستكون مهـمـة هذا البرنـامج إدارة هذه المدرسـة .. وعند التعمق أكثر ستجد أن هذه الطريقة في التفكير تقسم البرنـامج إلى مجموعـة كائنـات ، ستجد كائن المدير الذي سيؤدي مهـمـة ما ، ثم بعد ذلك كـائن المرشد الطلابي الذي سيؤدي مهـمـة ما ، ثم حاوية تحـوي كائنات الطلاب والذين سيؤدون مهـمـة ما ... إن هذه الطريقة في التفكير طريقة غير شيئيـة بل هي طريقة تفكير مشوهـة .. فلا هي هيكليـة ولا شيئيـة بل خليط لا طائل منـه سوى تشويـه أسلوب البرمجـة.

من جهـة أخرى سنجد أن طريقة التفكير الشيئي ستقسم البرنـامج إلى مجمـوعـة كائنـات ، سيكون هـناك كـائن المرشد الطلابي وحاوية تحـوي طلاب المدرسـة وستكون العلاقـة بين المرشد وهذه الحاويـة هي علاقـة صداقـة لأن من حق المرشد الطلابي الإطلاع على أسرار الطلبـة ، لاحظ أيضاً هـناك أنـه يوجد أكثر من حاويـة ، وبالتـالي فإن الطالب الواحد سيكون موجوداً أكثر من حاويـة ، هـناك حاويـة الفصل ، وحـاويـة للصف ، وحاويـة للمدرسـة ، عـند انتهـاء الـسنة الدراسيـة ستنتهي الفـصول والـصفوف وسيكون من الواجب إلغاؤهـا ... فهل سيكون من حق هذه الحاويات إلغاء الطلبـة وبالتـالي إلغاء المدرسـة ... لـذلك عليك التفكير عـن العلاقـة بين الحاويـة وعـناصرها .. هل سيكون من حق الحاويـة امتلاك عـناصرها وبالتالي إلغاءها أم لأ ... لاحظ أيضاً هـنا أن كـائن المعلم سيكون من حقه مراقبـة الطلبـة وتقييمهـم .... طريقة التفكير السابقـة لن تفكر بالعلاقات بين الكائنـات بهذا الـشكل .. وستحل المشاكل السابقـة "مشاكل الحاويات مثلاً" بطريقة هيكلية وليست شيئيـة .. لأن المبرمج الذي يفكر بطريقة هيكلية سيركز فقط على إيجاد الكائنـات وسيعتبر وجود الكائنـات كأنـه وجود للمتغيرات الأساسيـة فقط وبالتـالي فإنـه سيحل المـشكلة بطريقة هيكلية أما بالنسبة لطريقة التفكير الشيئيـة فهو سيركز أيضاً على إيجاد العلاقات بين الكائنـات .. لأن العالم يحتـوي علـى الكائنـات وعلـى العلاقات بينهـا ، وسيركز أيضاً على مزيـد من التجريد **Abstraction** لكـي يكون تركيزه على إيجاد حل لميدان المشكلـة (سنصل لهذه النقطـة لاحقاً).

معنى التجريد:

تعتبر لغـة التجميع تجريداً للغـة الآلـة ، فكلمـة **mov** هي تجريد لأمر التحريك الموجود في لغـة الآلـة ، حتى تفهم ميزة التجريد بشكل عام ، فلنقول أنك تريد برمجـة برنامج بواسطـة لغـة الآلـة ، تركيزك الأساسي هنا لـن يكون على إيجاد حل للمشكلـة التي تعـمل عليهـا فقط .. بل أيضاً علـى إيجـاد حلول لمـشاكل أخرى مثل مـشاكل الأخطـاء البشريـة عند إدخال الأوامر بصورتها الثنائيـة ، أيضاً يجب عليك معرفـة التفاصيل الداخليـة للجهاز الـذي تعـمل عليه ... أي يجب عليك التركيز على مشاكل أخرى غير ميدان المشكلـة الذي تعـمل عليـه ... لنفرض مرة أخرى أنـه طلب منك برمجـة نفس البرنامج ولكن هذه المرة بواسطـة لغـة التجميـع "الاسمبلي" ، أنـت تعلم أن هذه اللغـة هـي تجريد للغـة الآلـة ... وبالتالي فأنت لن تهتـم بشأن الأخطاء البشريـة لأنه بإمكانك كتابة أوامرك بواسطـة لغـة الإنسان ، هـذا الأمر سيجعلك تركز أكثر وأكثر على ميدان المشكلة الذي تريد حلـه وسيريحك بالتالي من التفكير فـي مشاكل أخرى .. صحيح أن لغـة الاسمبلي ليست ذات مستوى متقدم من التجريد وستضطر في النهايـة إلى التفكير ببعض المسائل غير ميدان مشكلة البرنـامج الذي تريد حلـه ولكن في النهاية فالأمر أفضل من البرمجـة بلغـة الآلـة .

لغة السي أتت أيضاً كتجريد للغة الاسمبلي ... في لغة الاسمبلي يجب عليك الاهتمام بالجهاز الذي تعمل عليه بينما هذا الأمر غير موجود في لغة السي ، فقط اكتب الكود الذي تريده ثم ترجمه على أي جهاز آخر وسيعمل لديك .. وهذا يعطيك مزيد من التفكير بشأن المشكلة التي تريد حلها في برنامجك ، ويقوم بإلغاء التفكير بشأن المشاكل الأخرى مثل ما هي التفاصيل الداخلية للجهاز الذي تعمل عليه.

نستطيع القول هنا أيضاً أن لغة الجافا هي تجريد آخر ، في لغة السي لا تستطيع أخذ الملف التنفيذي ونقله إلى جهاز آخر أو إلى نظام تشغيل آخر لأنه لن يعمل ... الملف التنفيذ الصادر عن لغة السي يعمل فقط على الجهاز الذي تمت ترجمة السورس كود عليه ، ويجب عليك الاهتمام هناك بتوفير مترجم للأجهزة الأخرى .. الأمر في لغة الجافا مختلف فلسفتها هي: **Write Once, run every where** ، أي اكتب السورس كود وسيعمل الملف التنفيذي على كل جهاز وعلى كل نظام تشغيل.

فكرة التجريد مطبقة بشكل كبيرة في البرمجة الشيئية .

انواع البيانات التجريدية Abstract Data Type (ADT).

في لغة الاكشن اسكربت توجد أنواع معرفة مسبقاً من قبل مترجم اللغة. مثلاً number يعتبر نوع نقوم بإستخدامه لمعالجة الارقام الصحيحة Manipulate ، والمقصود بكلمة معالجة هنا هو إجراء العمليات المعتادة على هذا النوع مثل "- + * /" وهكذا ، لأن الجمع والطرح والضرب عمليات تجرى عادة على الأرقام.

السلاسل الحرفية Strings نستعملها عند وجود الحاجة الى معالجة مصفوفات من نوع حرف. وعلى العكس من نوع number الذي نستطيع أن نقوم بعمليتي الضرب والقسمة عليه ، فطبيعة البيانات التي من نوع السلاسل الحرفية لايمكن ضربها أو قسمتها.

ADT هي قاعدة الاساس في البرمجة بطريقة OO ، وهي الخطوة الاولى التي يتم فيها تصميم وتعريف أنواع جديدة. هذه الانواع الجديدة يتم تعريفها وترجمتها من اوصاف وأفعال النوع نفسه، فالاوصاف يتم التعبير عنها بمتغيرات Variables ، و الأفعال يتم التعبير عنها بدوال Functions. مثلاً :

الشئ / الكائن	أوصاف	أفعال
سيارة	لون ، عدد الابواب ، لون ، سرعة ، نوع ، إسم ، ...	تشغيل ، إيقاف ، تغيير سرعات ، ...
صالون	لون ، نوع القماش ،عدد الكراسي ، ...	تركيب ، تنظيف ، فك...

الأوصاف والأفعال التي يتم تعريفها لهذه الأنواع الجديدة تختلف من برنامج الى أخر ، فمثلاً لو أردنا أن نقوم بتعريف نوع سيارة لإستخدامه في برنامج تحكم بالسرعات فإن لون السيارة سيكون غير مهم بالنسبة لنا.

يسمى هذا النوع من البيانات بالبيانات التجريدية وذلك لأنها لازالت تحتاج الى تعريف محدد لإستخدامها من قبل لغات البرمجة ، فنوع القماش مثلاً عبارة عن معلومة يجب ترجمتها حتى يمكن التعبير عنها ، فمثلاً :

```
number Fabric;  // 1=Blue, 2=Red….
```

او

```
String Fabric; // B=blue, R=red, …
```

الكائنات والكبسلة:

للبرمجة الشيئية فلسفتها القائمة على تقسم البرنامج ليس إلى مهام كما هو الحال في البرمجة الهيكلية بل إلى أجسام أو كائنات في العالم الفيزيائي الحقيقي ، مفاهيم الكائنات هي على كل حال ما يلي:

الفئات classes :

تساعد الفئة على تمييز الكائن من خلال تعريفها ، إذا نظرنا مثلاً إلى فئة اسمها سائق ، ستجد أن هناك فرق بين السؤال: كيف يقود السائق ، وفرقاً بين السؤال: كيف يقود السائق أحمد ، الكائن أحمد هو سائق سباق سيارات رالي.

عند السؤال عن ما هو السائق، فإنك في الحقيقة تتكلم عن حالة عامة ، كل السائقين يعلمون كيفية تشغيل السيارة ، وكل السائقين لديهم رخصة قيادة (إلا في حالات شاذة طبعاً) ، وكل السائقين لديهم مخالفات

مرورية ، وكل السائقين لديهم سيارة أيضاً؛ باختصار فإن الفئة class تقوم بكبسلة تصنيف معين لمجموعة من الكائنات objects .. وهو في مثالنا هذا كائنات السائقين.
ولكن حينما نتحدث عن سائق معين فنحن هنا ننتقل من مفهوم الفئة class إلى مفهوم الكائن object ...حينما نتحدث عن الكائن أحمد في المثال السابق ، فإنه وبسبب معرفتنا بأنه ينتمي إلى فئة السائقين فإننا نعرف أنه يملك رخصة قيادة ولديه سيارة ، ما لا نعرفه ربما بعض السمات والخصائص المتواجدة لدى السائق أحمد ... كم سيارة لديه ؟ ، ما مدى مهاراته في قيادة السيارات؟ ، ما نوع الرخصة المرورية التي لديه ... وبالتالي فإن الكائن object هو حالة خاصة instance من الفئة.
الخلاصة :ان الوراثة Inheritance.
هي إمكانية توريث صفات وأفعال Class معينة الى Class اخرى جديدة ، وذلك لتجنب عملية تكرار الصفات المشتركة بين الـ classes. حيث تسمى الـ Class الجديدة بالابن والاولى بالأب.

المكونات Components :

التفكير في الكائن عن طريق مكوناته الصغيرة أمر ضروري وأساسي للغاية في البرمجة الشيئية ، فئة السيارة مثلاً تتكون من المحرك وعجلة القيادة ومخزن البنزين والإطارات ... إلخ ؛ المكون في النهاية هو عبارة عن فئة class ولكنها أقل حجماً وأكثر تحديداً ، فئة السيارة إذا تم وصفها في الكود على أنها سيارة فقط فسيكون الكود أقل مقروئية ، ولكن بدلاً من ذلك ماذا لو تم وصف فئة السيارة على أنها فئة تدير وتتعامل مع مكونات أقل حجماً ، هذا سيسهل كثيراً من قراءة الكود ومن تصميم الكود ويجعلك قادراً على فهم الفئة class بشكل أوضح كما هي في العالم الحقيقي ، لاحظ أيضاً هنا أنه من الممكن أن تحتوي هذه المكونات على مكونات أخرى.

الخواص Properties:

السمات أو الخواص هو ما يفرق الكائنات عن بعضها البعض ، إذا نظرنا إلى العالم الحقيقي سنجد أن ما يفرق بين إنسان وإنسان آخر هو شكل الوجه مثلاً والطول وحتى العقيدة ومجموعة القيم التي تكون الإنسان والأخلاق .. وغير ذلك من الخواص الخاصة بكل إنسان.
بإمكانك التفكير عن الخواص في مستوى الفئات classes ، فمثلاً في مثال السائق ، سنجد أن جميع السائقين يجب أن يكونوا يملكون رخصة قيادة ، وبالتالي فهذه خاصية مشتركة وعامة بين جميع الكائنات من نوع فئة السائق.

السلوكيات Behaviors:

بإمكاننا التفكير عن السلوك على مستوى الكائن ومستوى الفئة ، في مثال السائق سنجد أن هناك سلوك اسمه: يقود (الموقع الذي تريده) ، وسنجد أن جميع الفئات ستشترك في فعل القيادة هذا ، ولكن الاختلاف هنا يمكن بين كل كائن وكائن ، فإذا كان الكائن أحمد سريعاً للغاية فإنه سيصل في وقت أسرع ولكن احتمالات وقوع حادث كبيرة ، بينما الكائن محمد سائق هادئ نوعاً ما فإنه سيصل في وقت أقل ولكن بدرجة أمان أكبر من صاحبه السابق.

الكبسلة Encapsulation:

الفئة والكائن تعتبران في الأساس تغليفاً للسلوكيات والخصائص في مجموعة واحدة أو قالب واحد ، هذه الكبسلة والتغليف ستسهل عليك البرمجة بشكل كبير فأنت الآن ستكون قادراً على التعامل مع هذه الكبسولة بشكل شامل ، فأي تغيير في إحدى خصائص كائن ما ، سيقوم بتغيير جميع خصائص هذا الكائن ، لنأخذ مثالاً واقعياً لنفرض أن الطالب أيمن رسب في اختبار إحدى المواد ، سيؤثر هذا على معدله العام وليس ذلك فحسب بل سيؤثر على خصائصه المتبقية فهو أولاً سيتأثر نفسياً وبالتالي سيؤثر هذا الأمر على حالته الجسدية العامة ، لكن إذا نظرنا إلى البرمجة الهيكلية فحينما تقوم بجعل درجة الطالب أحمد درجة سيئة ستكون مضطراً على تغيير كل خاصية من خصائصه مما سيجعل الكود طويلاً وليس ذا مقروئية عالية ، لكن الأمر يختلف في البرمجة الشيئية فبواسطة سطر واحد من الكود تستطيع فعل كل هذه التغييرات .
الخلاصة: ان الكبسلة "التجميع والتغليف": وهي تمثل عملية تجميع صفات وأفعال شئ معين ووضعها داخل مغلف وأخيراً تسميتها بأسم هذا الشئ ... مثلاً الـ Stack ، له صفات "Attributes" وهي حجم ، ونوع ، وحيز ... أيضاً الـ Stack له عمليات أو أفعال "Operations or Methods" يقوم بها وهي Pop و Push.

إخفاء البيانات Data Hiding :

من المفيد أن يكون هناك تجزئة في قطاع البرمجة إلى صانعي الفئات **class creator** وإلى مستخدمي الفئات أو المبرمجين الزبائن **client programmers** ، النوع الأول من المبرمجين هو من يقوم بإنشاء الفئات والنوع الثاني هو من يستخدم هذه الفئات ويجمعها لكي يبني من خلالها نظاماً أو برنامجاً متكاملاً . الوظيفة الأساسي لصانعي الفئات فهي بناء فئة يكشف فيها للمبرمج الزبون ما هو ضروري فقط، لماذا؟ .. حتى يسمح للمبرمج الزبون بالتفكير في ميدان المشكلة دون التورط في التفكير بشأن مسائل أخرى.

بالرغم من الفائدة الضرورية لعمل صانه الفئات إلا أنه سيفرض قيود على المبرمج الزبون عند استخدام فئة ما، فهو لا يستطيع الوصول إلى الأعضاء المخفيين **private** .. لكن إذا فكرت قليلاً ستجد أن هذه الأجزاء الداخلية مهمة للعمل الداخلي داخل الفئة وليس لها أي علاقة بتاتاً بالأعضاء العامة **public** أو واجهة الفئة .... في مثال السائق الذي يوجد فيه سلوك (أو دالة إن شئت) اسمها يقود ، ستجد أنك تستخدم هذا السلوك دون أي تفكير مسبق في كيف سيعمل هذا السلوك ، لمثال أوضح أنظر إلى هذا المثال المكتوب بواسطة لغة السي شارب:

```
Console.Write (" O O P\n");
```

هذا السطر يقوم بطباعة العبارة **O O P** ، لاحظ هناك أنك لا تهتم بما يحدث داخل الدالة **Write** بل كل ما فكرت فيه هو المشكلة التي تود حلها فقط وحينما احتجت طباعة عبارة ما ... فإنك قمت بفتح التوثيقات الخاصة بلغة السي شارب ووجدت أنه يوجد فئة اسمها **Console** لديها دالة ساكنة اسمها **Write** تقوم بطباعة أي كلام تريده على الشاشة ... ثم استخدمتها ولم تحتج بتاتاً لأي عمليات معقدة تقوم بها، لأن هذه العمليات المعقدة تحدث داخلياً داخل الفئة **Console** ، مثل هذا الأمر سيجعلك تفكر أكثر في المشكلة التي تريد حلها وليس في المشاكل الصغيرة التي ستتولد أثناء محاولتك حل المشكلة الكبيرة.

الخلاصة: **ان إخفاء البيانات : وهي عملية إخفاء المتغيرات والدوال ، بالنسبة للمتغيرات يجب دائماً إخفائها عن طريق تحديدها بـ private اما الدوال فذلك يرجع لإحتياج مصمم البرنامج ، حيث يوجد ثلاث أنواع من الإخفاء public, protected and private ، سنؤجل شرح عمل الـ protected لعلاقتها بالوراثة.**

- تعددية التشكل Polymorphism.

تعدد التشكل وهي منح المبرمج إمكانية تعريف دالتين أو أكثر بنفس الأسم ، ولكن يشترط أن تختلف كل من هذه الدوال في قائمة إرسال/إستقبال المتغيرات Parameters ، أياً كان هذا الإختلاف في العدد أو النوع أو كليهما.

التماسك والتزاوج Cohesion and Coupling :

نأتي الآن إلى أمر مهم يندر الحديث عنه رغم أهميته ولو أنه يتردد بكثرة في مواضيع هندسة البرمجيات ، وهو موضوع التماسك.

التماسك: هو قدرة المقطع الواحد على تنفيذ مهمة واحدة من دون الحاجة إلى استدعاء مقاطع برمجية أخرى.

من المعلوم أن أي برنامج يتم تقسيمه إلى مقاطع برمجية صغيرة ، وهو يقسم إما إلى فئات أو دوال أو إلى أي شيء آخر ... وهذا التقسيم هو ما يقصد به المقطع الواحد في التعريف السابق.

التزاوج هو عكس التعريف السابق ، وهو يعني أن أكثر من مقطع برمجي مرتبطة مع بعضها بشكل يجعلهم لا يستطيعون العمل إلا أثناء تواجدهم كلهم مع بعضهم في برنامج واحد.

من التعريفين السابقين نجد أنه كلما ازداد التماسك في الفئات قل التزاوج فيما بينهم ، هناك تقريباً سبعة أنواع من التماسك أقواها هو التماسك الوظيفي والذي يعني قدرة المقطع على أداء مهمة واحدة فقط من دون الاعتماد على أي مقطع برمجي آخر.

أما أضعفها فهو **Coincidental Cohesion** وهو حينما يقوم المقطع البرمجي بأداء أكثر من مهمة غير مترابطة.

لكي نطلق على لغة برمجة معينة بأنها تدعم البرمجة بطريقة OO يجب أن تتوفر فيها التقنيات الأتية:

١- انواع البيانات التجريدية Abstract Data Type (ADT).
٢- الكبسلة Encapsulation.
٣- إخفاء البيانات Data Hiding.
٤- تعددية التشكل Polymorphism.
٥- الوراثة Inheritance.

الان لنبدأ الدراسة العملية ل Object-Oriented Programming with ActionScript 2.0

ActionScript Classes

سوف نرمز للClasses فى حديثنا بالخلايا و احيانا قد تجدنى اكتبها كلاس بالعربية و ذلك للتسهيل و للتعود على المصطلح الاجنبى .

Class

وهي مكوّن أساسيّ لبناء الكود، ليس لها واجهة مرئيّة، ولكنّها تؤّدى وظائف معيّنة لبرنامجك.
وباختصار، فإنّ الخليّة هي مجموعةٌ من الدوالّ والإجراءات التي يمكنُ إعادة استخدامها في أيّ مشروع، بحيثُ يوفّر المبرمجُ على نفسِه مشقّة إعادة كتابتها مرة أخرى.
لاحظ ان الخليّة هي برنامج لا يمكن تنفيذه بمفرده، إذ يجب أن يتمّ استخدامه بواسطة تطبيق آخر أو جزء آخر من نفس التطبيق الذي يحتوي على الخليّة.
ستسألني:

- أتعني أنّنا نستدعي الخليّة كما نستدعي الدوال والإجراءات الفرعيّة، باستخدام اسم الدالة.
- إطلاقا.. إنّ الخليّة تحتوي أساسا على العديد من الإجراءات والدوال والخصائص الخاصّة بها.
- إذن كيف نستخدم الخليّة؟
- أنّنا نعرّف متغيّرا من نفس نوع الخليّة، تماما كما نعرّف متغيرا من نفس نوع البيانات التى ينتمى لها المتغير .

- بالضبط.. وهذا يعني ببساطة، أنّ الخليّة هي إحدى أنواع المتغيرات التي يعرّفها المستخدم User Defined Types، وإن كانت هي أشملها وأقواها وأهمّها على الإطلاق، كما سنرى بعد قليل.

الآن تدرك بالطبع أنّك أوغلت في استخدام الخلايا، فهي ليست جديدة أو غريبة.. الجديد هو أنّك ستتعلّم هنا كيف تبني الخلايا الخاصّة بك.

- بقى ان تعرف ان الخلايا يتم تعريفها فى ملفات مستقلة و ليست فى الtime line و تكون هذة الملفات بامتداد as و يمكنك كتابة الكود الخاص بها فى المحرر الخاص بذلك الموجود فى برنامج فلاش النسخة الاحترافية او باستخدام النوت باد و حفظ الملف بامتداد as .
- لاحظ ايضانة يجب ان يكون اسم ملف الخلية هو ذات اسم الخلية مثلا خلية اسمها Product يجب ان يكتب الكود الخاص بها فى ملف اسمةProduct.as و يكون هذا الملف فى نفس المجلد الذى سوف يحفظ فية ملف الفلاش الذى سوف يستخدم فية الخلية Product و يمكن ايضا ان يحفظ ملف الخلية فى مجلد فرعى داخل المجلد الرئيسى مثلا نقوم بعمل مجلد فرعى و ليكن اسمة Data و نحفظ فية ملف الخلية .

و للأستدعاء الخلية نقوم بكتابة الكود التالى فى ملف الفلاش :

```
var myProduct:Product = new Data.Product();
import Data.Product;
```

مثال يوضح طريقة كتابة الخلية :

```
class Product {


    var id:Number;

    var prodName:String;

    var description:String;


    function Product (id:Number, prodName:String, description:String)

    {

        this.id = id;

        this.prodName = prodName;

        this.description = description;

    }

}
```

لاحظ Public and Private Attributes

Public: ليصبح الخلية متاحا للاستخدام من أي موضع في مشروعك، أو من أي مشروع آخر يضيف مرجعا لمشروعك**.**

Private: ليصبح الإجراء متاحا للاستخدام، فقط من داخل الدالة أو الخليّة Class .

The extends Keyword

تستخدم لعمل وراثة لخصائص و دوال خلايا اخرى سبق تعريفها وهى public مثال يمكن وراثة الخلية MovieClip و استخدام كل خصائصها **فى عمل خلية جديدة و ليكن اسمها** class Drag **و فى هذة الحالة الخلية الجديدة يمكنها استعمال كلا من**

onPress(), onRelease(), startDrag(), and stopDrag(),

و كل ذلك بسبب وراثة الخلية MovieClip class .

```
classclass Drag extends MovieClip {


    function Drag () {

        onPress = doDrag();

        onRelease = doDrop();

    }

    private function doDrag(){

        this.startDrag();

    }

    private function doDrop(){

            this.stopDrag();

    }

}
```

- **الان يمكن ربط خلية ما بعنصر على مسرح العمل مثل موفى كليب من خلال استخدام خاصية linkage مثال :**
- **مثال : قم بعمل دائرة ثم اضغط f8 لتحويلها لموفى كليب فتظهر لك هذة الصورة :**

فقم باختيار Export for ActionScript

و فى مربع النص **AS 2.0 class** اكتب اسم الخلية التى تريد ربطها مع هذا الموفى كليب و ليكن مثلا **Drag.as**

الان عندما تستخدم هذا الموفى كليب سيكون لة كل خصائص الخلية **Drag.**

رسائل الخطأ Error Messages

يقدم برنامج فلاش مجموعة من رسائل الخطأ و منها التالى :

Error number	Message text
1093	A class name was expected.
1094	A base class name is expected after the 'extends' keyword.
1095	A member attribute was used incorrectly.
1096	The same member name may not be repeated more than once.
1097	All member functions need to have names.
1099	This statement is not permitted in a class definition.
1100	A class or interface has already been defined with this name.
1101	Type mismatch.
1102	There is no class with the name '<ClassName>'.
1103	There is no property with the name '<propertyName>'.
1104	A function call on a non-function was attempted.
1105	Type mismatch in assignment statement: found [lhs-type] where [rhs-type] is required.
1106	The member is private and cannot be accessed.
1107	Variable declarations are not permitted in interfaces.
1108	Event declarations are not permitted in interfaces.
1109	Getter/setter declarations are not permitted in interfaces.
1110	Private members are not permitted in interfaces.
1111	Function bodies are not permitted in interfaces.
1112	A class may not extend itself.
1113	An interface may not extend itself.
1114	There is no interface defined with this name.
1115	A class may not extend an interface.
1116	An interface may not extend a class.
1117	An interface name is expected after the 'implements' keyword.
1118	A class may not implement a class, only interfaces.

Error number	Message text
1119	The class must implement method 'methodName' from interface 'interfaceName'.
1120	The implementation of an interface method must be a method, not a property.
1121	A class may not extend the same interface more than once.
1122	The implementation of the interface method doesn't match its definition.
1123	This construct is only available in ActionScript 1.0.
1124	This construct is only available in ActionScript 2.0.
1125	Static members are not permitted in interfaces.
1126	The expression returned must match the function's return type.
1127	A return statement is required in this function.
1128	Attribute used outside class.
1129	A function with return type Void may not return a value.
1130	The 'extends' clause must appear before the 'implements' clause.
1131	A type identifier is expected after the ':'.
1132	Interfaces must use the 'extends' keyword, not 'implements'.
1133	A class may not extend more than one class.
1134	An interface may not extend more than one interface.
1135	There is no method with the name '<methodName>'.
1136	This statement is not permitted in an interface definition.
1137	A set function requires exactly one parameter.
1138	A get function requires no parameters.
1139	Classes may only be defined in external ActionScript 2.0 class scripts.
1140	ActionScript 2.0 class scripts may only define class or interface constructs.
1141	The name of this class, '<A.B.C>', conflicts with the name of another class that was loaded, '<A.B>'. (This error occurs when the ActionScript 2.0 compiler cannot compile a class because of the full name of an existing class is part of the conflicting class' name. For example, compiling class mx.com.util generates error 1141 if class mx.com is a compiled class.)
1142	The class or interface '<Class or Interface Name>' could not be loaded.
1143	Interfaces may only be defined in external ActionScript 2.0 class scripts.

Error number	Message text
1144	Instance variables cannot be accessed in static functions.
1145	Class and interface definitions cannot be nested.
1146	The property being referenced does not have the static attribute.
1147	This call to super does not match the superconstructor.
1148	Only the public attribute is allowed for interface methods.
1149	The import keyword cannot be used as a directive.
1150	You must export your Flash movie as Flash 7 to use this action.
1151	You must export your Flash movie as Flash 7 to use this expression.
1152	This exception clause is placed improperly.
1153	A class must have only one constructor.
1154	A constructor may not return a value.
1155	A constructor may not specify a return type.
1156	A variable may not be of type Void.
1157	A function parameter may not be of type Void.
1158	Static members can only be accessed directly through classes.
1159	Multiple implemented interfaces contain same method with different types.
1160	There is already a class or interface defined with this name.
1161	Classes, interfaces, and built-in types may not be deleted.
1162	There is no class with this name.
1163	The keyword '<keyword>' is reserved for ActionScript 2.0 and cannot be used here.
1164	Custom attribute definition was not terminated.
1165	Only one class or interface can be defined per ActionScript 2.0 .as file.
1166	The class being compiled, '<A.b>', does not match the class that was imported, '<A.B>'. (This error occurs when a class name is spelled with a different case from an imported class. For example, compiling class mx.com.util generates error 1166 if the statement import mx.Com appears in the util.as file.)
1167	You must enter a class name.
1168	The class name you have entered contains a syntax error.

Error number	Message text
1169	The interface name you have entered contains a syntax error.
1170	The base class name you have entered contains a syntax error.
1171	The base interface name you have entered contains a syntax error.
1172	You must enter an interface name.
1173	You must enter a class or interface name.
1174	The class or interface name you have entered contains a syntax error.
1175	'variable' is not accessible from this scope.
1176	Multiple occurrences of the 'get/set/private/public/static' attribute were found.
1177	A class attribute was used incorrectly.
1178	Instance variables and functions may not be used to initialize static variables.
1179	Runtime circularities were discovered between the following classes: <list of user-defined classes>. This runtime error indicates that your custom classes are incorrectly referencing each other.
1180	The currently targeted Flash Player does not support debugging.
1181	The currently targeted Flash Player does not support the releaseOutside event.
1182	The currently targeted Flash Player does not support the dragOver event.
1183	The currently targeted Flash Player does not support the dragOut event.
1184	The currently targeted Flash Player does not support dragging actions.
1185	The currently targeted Flash Player does not support the loadMovie action.
1186	The currently targeted Flash Player does not support the getURL action.
1187	The currently targeted Flash Player does not support the FSCommand action.
1188	Import statements are not allowed inside class or interface definitions.
1189	The class '<A.B>' cannot be imported because its leaf name is already resolved to the class that is being defined, '<C.B>'. (For example, compiling class util generates error 1189 if the statement import mx.util appears in the util.as file.)
1190	The class '<A.B>' cannot be imported because its leaf name is already

Error number	Message text
	resolved to a previously imported class '<C.B>'. (For example, compiling import jv.util generates error 1190 if the statement import mx.util also appears in the AS file.)
1191	A class' instance variables may only be initialized to compile-time constant expressions.
1192	Class member functions cannot have the same name as a superclass' constructor function.
1193	The name of this class, '<ClassName>', conflicts with the name of another class that was loaded.
1194	The superconstructor must be called first in the constructor body.
1195	The identifier '<className>' will not resolve to built-in object '<ClassName>' at runtime.
1196	The class '<A.B.ClassName>' needs to be defined in a file whose relative path is <'A.B>'.
1197	The wildcard character '*' is misused in the ClassName '<ClassName>'.
1198	The member function '<classname>' has a different case from the name of the class being defined, '<ClassName>', and will not be treated as the class constructor at runtime.
1199	The only type allowed for a for-in loop iterator is String.
1200	A setter function may not return a value.
1201	The only attributes allowed for constructor functions are public and private.
1202	The file 'toplevel.as', which is required for typechecking ActionScript 2.0, could not be found. Please make sure the directory '$(LocalData)/Classes' is listed in the global classpath of the ActionScript Preferences.
1203	Branch between <spanStart> and <spanEnd>> exceeds 32K span.
1204	There is no class or package with the name '<packageName>' found in package '<PackageName>'.
1205	The currently targeted Flash Player does not support the FSCommand2 action.
1206	Member function '<functionName>' is larger than 32K.
1207	Anonymous function around line <lineNumber> exceeds 32K span.
1208	Code around line <lineNumber> exceeds 32K span.
1210	The package name '<PackageName>' cannot also be used as a method

Error number	Message text
	name.
1211	The package name '<PackageName>' cannot also be used as a property name.
1212	The ASO file for the class '<ClassName>' could not be created. Please make sure the fully-qualified class name is short enough so that the ASO filename, '<ClassName.aso>', is less than 255 characters.
1213	This type of quotation mark is not allowed in ActionScript. Please change it to a standard (straight) double quote.

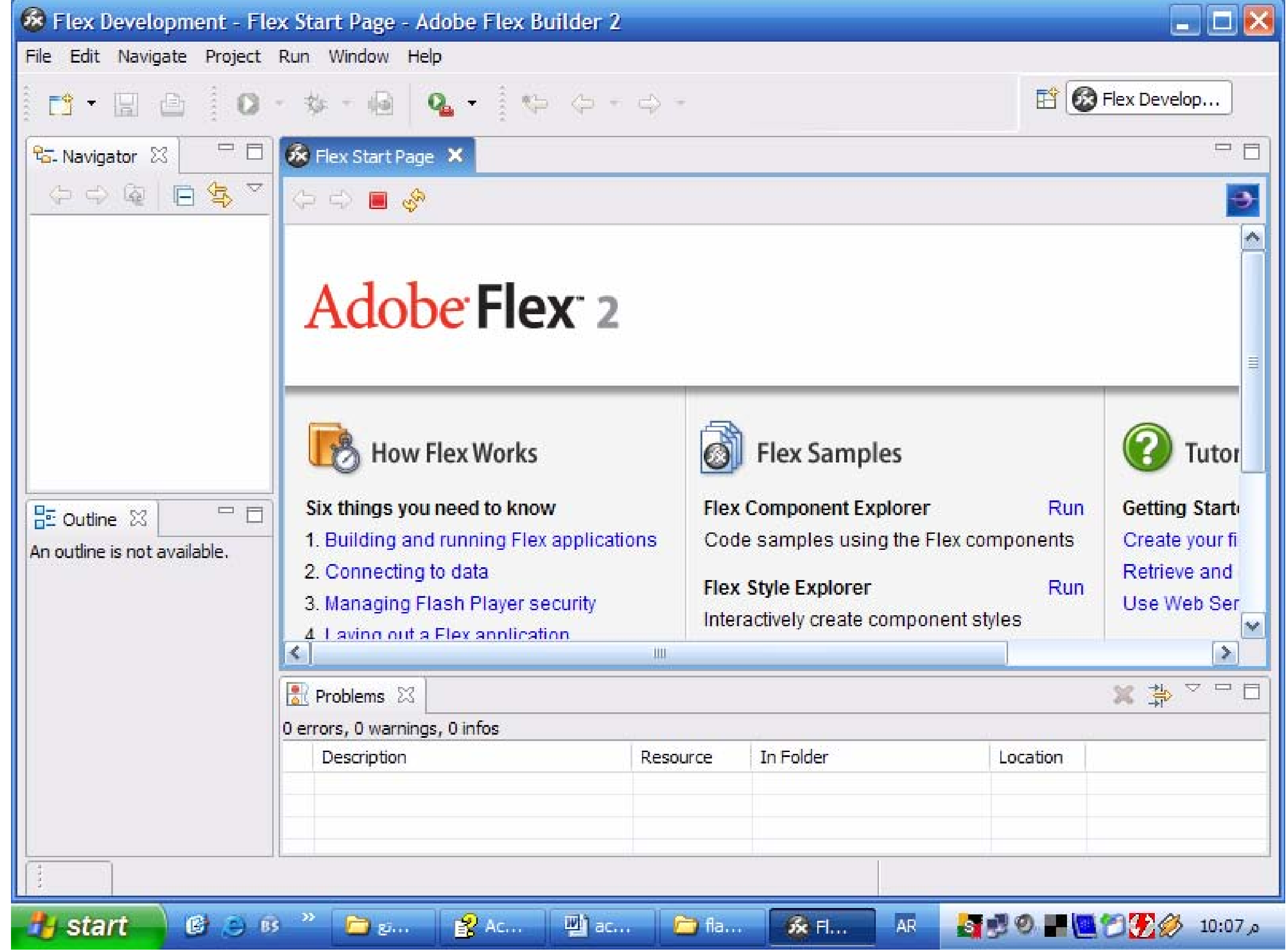
الفصل السادس عشر برنامج Flex 2.0

مقدمة عن التطور التاريخى لبرنامج الفلاش مع المشغل الخاص بكل اصدار للبرنامج

Flash Player 9

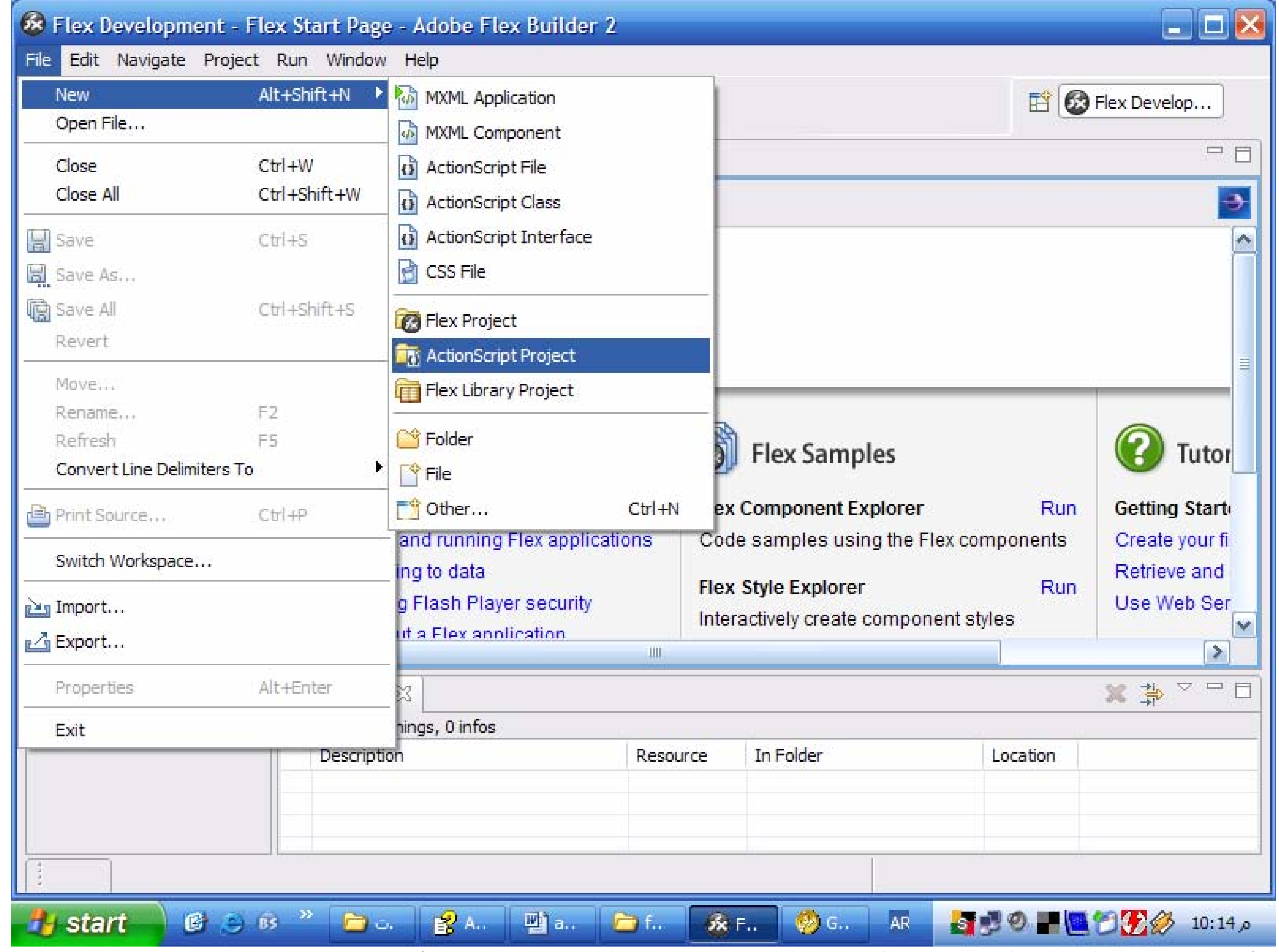
- 1997 - FutureSplash is renamed and released as Macromedia Flash 1.0.
- 1997 - Flash 2 adds buttons, the Library, sound support, and color tweens.
- 1998 - Flash 3 adds new animations and alpha transparency.
- 1999 - Flash 4 adds MP3 audion streaming and motion tweens.
- 1999 - 100-millionth download of the Flash Player.
- 2000 - Flash 5 adds ActionScript 1.
- 2000 - Flash Player installs reach the 92 percent penetration mark.
- 2002 - Flash Player 6 adds support for Flash remoting, Web services, video, shared libraries, and components.
- 2003 - Flash Player 7 introduces ActionScript 2.0 and streaming audio and video.
- 2005 - Flash Player 8 adds filter effects, GIF and PNG images, bitmap caching, new video codec (On2 VP6), file upload and downloading, and FlashType.
- 2006 - Flash Player 9 is released and introduces ActionScript 3, E4X XML parsing, regular expressions, and binary sockets.

عندما نقوم بفتح البرنامج **Flex 2.0** تظهر لنا الشاشة التالية

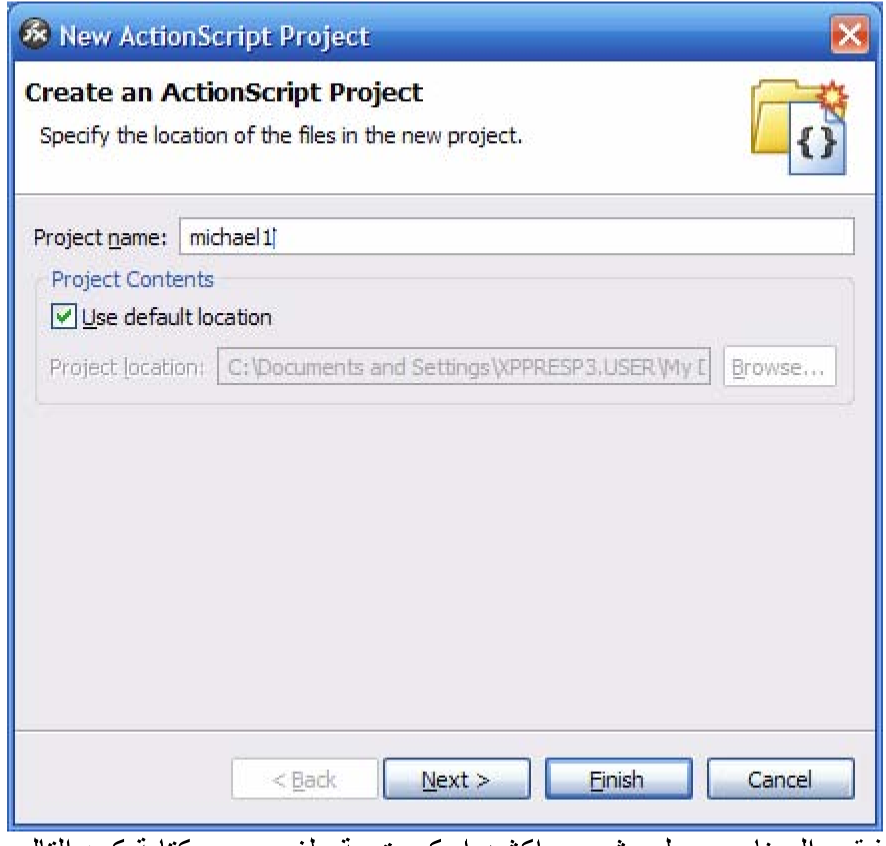


تلاحظ ان البرنامج يعرض فى البداية طريقة عملة و امثلة خاصة بة و مشاريع و مواقع على الانترنت

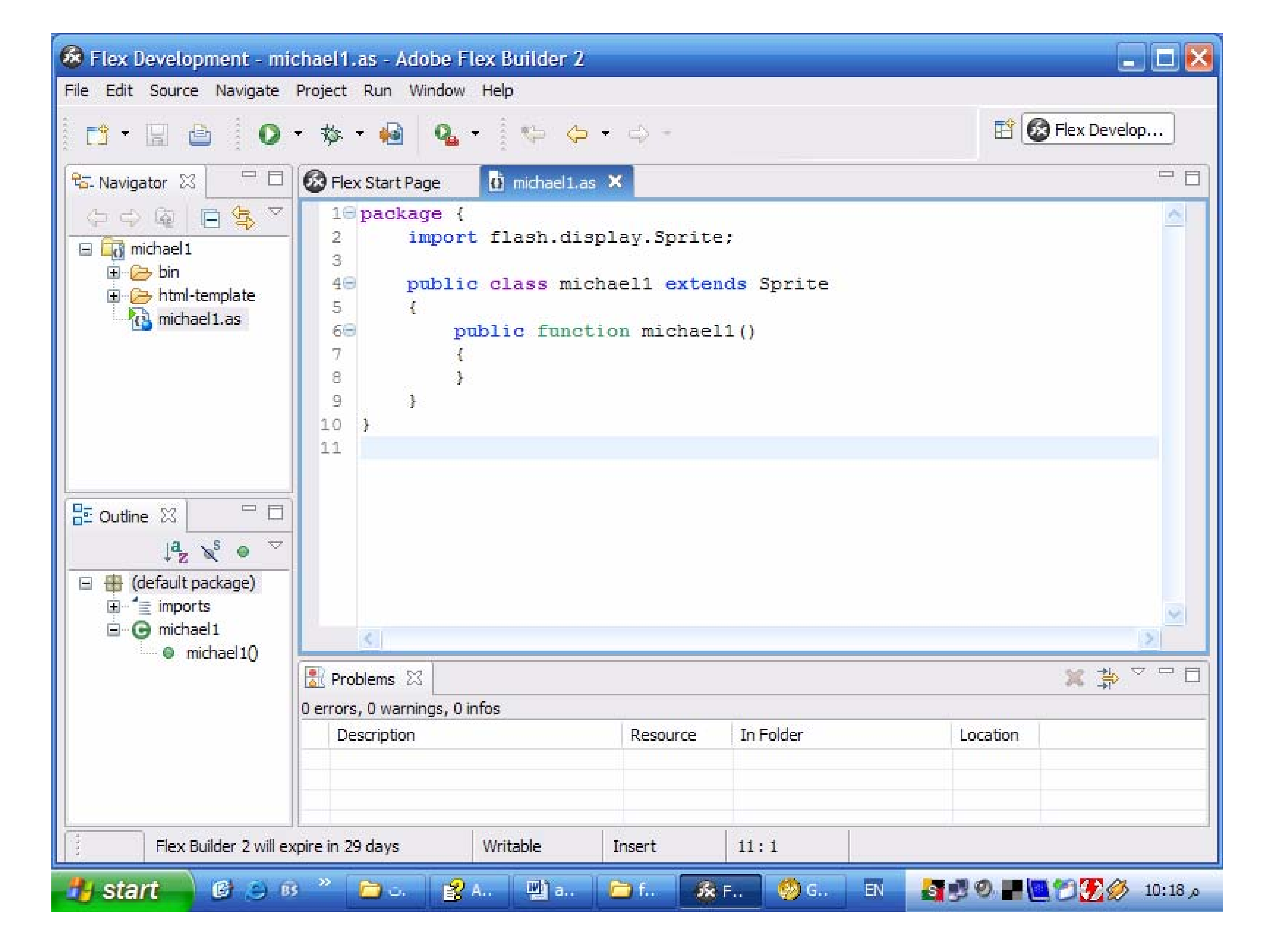
و لعمل مشروع جديد اضغط على File New ActionScript Project



يظهر لك شاشة المعالج السحرى و كل ما عليك هو كتابة اسم المشروع و الضغط على زر Finish



فيقوم البرنامج بعمل مشروع اكشن اسكربت بة ملف as و كتابة كود التالى



لاحظ ان

ActionScript 3.0 is completely class-based, so all code must be placed in methods of your project's Classes

اى ان لم يعد من الممكن كتابة الكود فى اى مكان داخل البرنامج بل اصبح المشروع عبارة عن كلاس كبير داخلة مجموعة من الوظائف كمثال السابق عندما تقوم بعمل مشروع جديد تلقائي يقوم البرنامج بعمل كلاس كودها كالتالى

```
package {
    import flash.display.Sprite;

    public class michael1 extends Sprite
    {
        public function michael1 (  )
        {

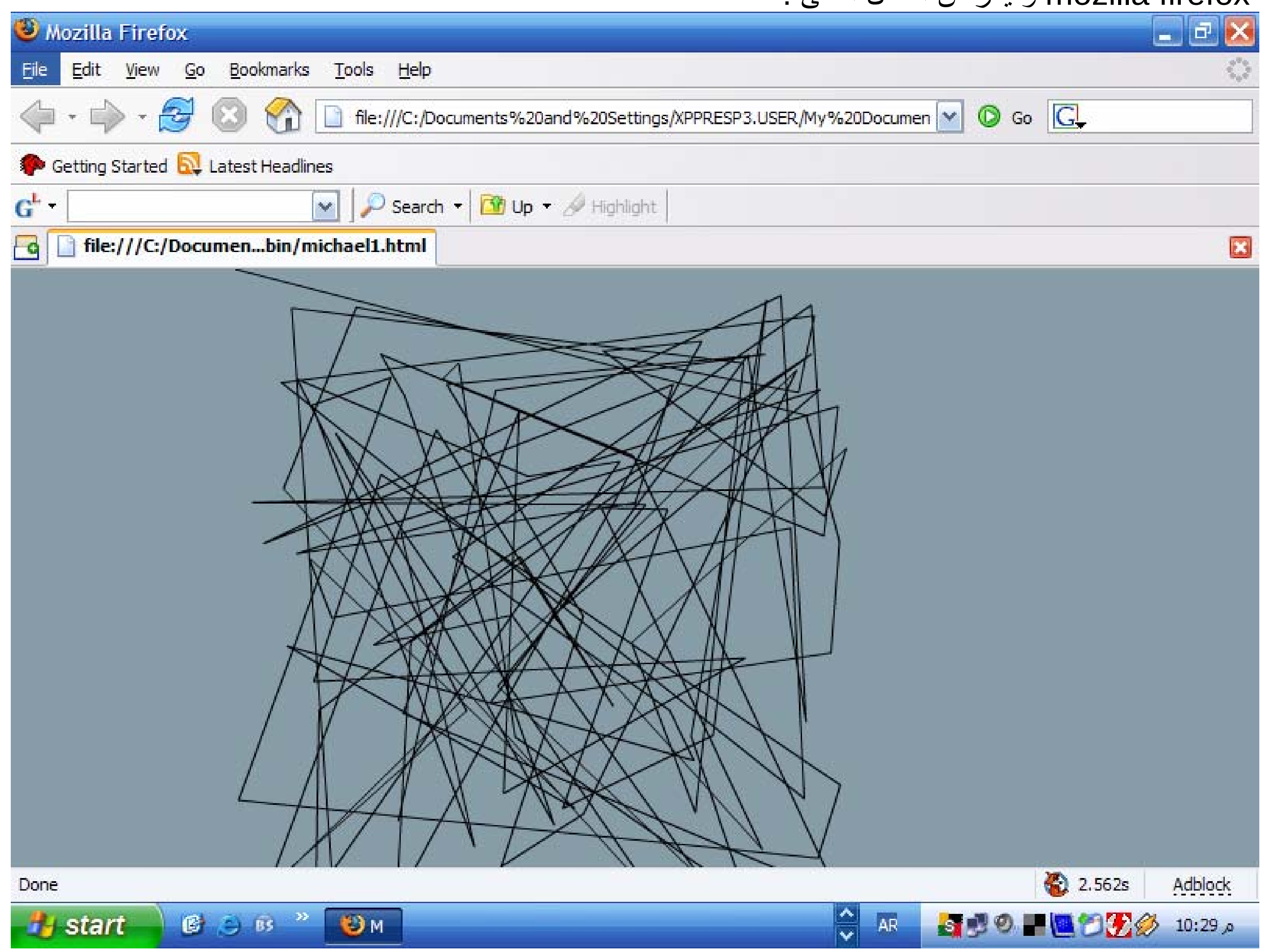
        }
    }
}
```

و انت تقوم بكتابة كود البرنامج الخاص بك داخل الدالة public function michael1 مثال

```
package {
    import flash.display.Sprite;
```

```
    public class michael1 extends Sprite {
        public function michael1 (  ) {
            graphics.lineStyle(1, 0, 1);
            for(var i:int=0;i<100;i++) {
                graphics.lineTo(Math.random(  ) * 400,
Math.random(  ) * 400);
            }
        }
    }
}
```

و النتيجة هى رسم ١٠٠ خط عشوائي و يظهر ذلك عند الضغط على زر run فيقوم البرنامج بتشغيل متصفح mozilla firefox و يعرض الشكل التالى :



و لكى تتبع طريقة سير البرنامج هل هى صحيحية ام لا استخدم الكود التالى

```
package {
    import flash.display.Sprite;

    public class michael1 extends Sprite {
        public function michael1(  ) {
            var userName:String Michael nabil";
            trace("My name is " + userName + ".");
        }
    }
```

}

The elements of a Flex application

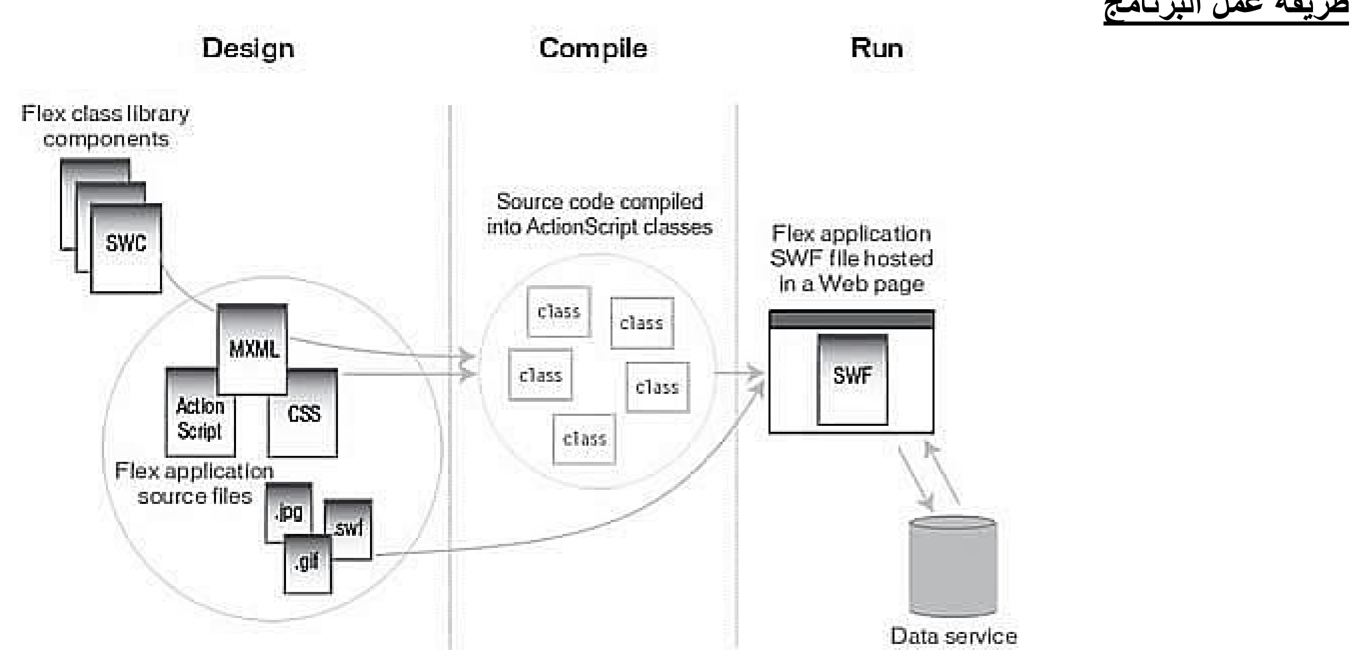
العناصر التى يتكون منها المشروع داخل برنامج Flex

اى مشروع يجب ان يحتوى على العناصر التالية :

Flex framework	بيئة اطار العمل الخاصة ببرنامج The Adobe® Flex 2 و التى تحتوى على كل الكمبونات او الادوات اللازمة لبناء موقع على الانترنت او برنامج بصفة عامة و المقصود ان البرنامج يوفر لك واجهة مليئة بالادوات المستخدمة فى جمع البيانات من المستخدم مثل مربعات الحوار و الازرار و اداوات عرض البيانات و وسائل التأكد من صحة الادخال و كثير من المؤثرات و كل هذا يرتبط بالمكتبة component library (SWC) file .
MXML	كل مشروع يحتوى على الاقل ملف واحد MXML file معرف فى بداية المشروع لاحظ ان ملف MXML يعتبر لغة قريبة فى تركيبها من لغة ال html و لغة ال xml بالطبع . و هذة اللغة صممت خصيصا لهذا البرنامج لوصف بناء المشاريع من خلال الوسوم tags.
ActionScript 3.0	لكى تقوم بعمل تطبيقات تتفاعل مع المستخدم يجب ان تستخدم لغة ActionScript 3.0 و هى شبيهة فى تكوينها من لغة الجافا اسكربت و يمكنك اضافة اكواد الاكشن اسكربت ٣,٠ داخل البرنامج مباشرة من خلال ملف ال MXML بين tags او فى ملف منفصل و عمل استيراد لة داخل المشروع .
CSS	جداول الانماط المتعاقبة و المقصود بها عمل شكل خاص بالازرار او الادوات المستخدمة فى النماذج
Graphic assets	مثل معظم البرامج يمكنك استخدام صور و ايقونات و مؤثرات
Data	بعض الادوات تستخدم لعرض البيانات مثل (a combo box or data grid for example) و ذلك من خلال المصفوفات او مصدر خارجى للبيانات مثل ملفات النصوص txt او xml

How Flex applications are compiled and deployed

طريقة عمل البرنامج



يمكنك عمل النماذج الخاصة بالبرنامج الخاص بك من خلال كود mxml و هذا مثال لعمل لوحة بها نص و اضا زر للأغلاقها:

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml" layout="absolute">
<mx:Panel>
<mx:TextArea text="Say hello to Flex!" />
<mx:Button label="Close" />
</mx:Panel>
</mx:Application>
```

اذا كنت معتاد على لغة ال xml فسوف تلاحظ التنسق الخاص بهذةاللغة و يمكنك الرجوع الى الملحق الخاص بلغة الxml لكى تفهم الفكر الاساسى لهذة اللغة ..

المهم ان كل مشروع سوف تقوم بعملة داخل برنامج flex2

سوف يحتوى فى المقدمة على كود

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml" layout="absolute">
```

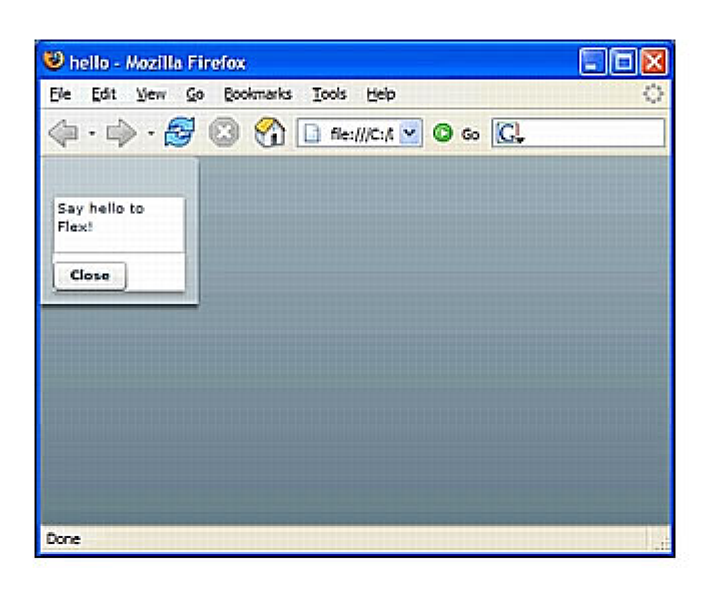
و يجب ان ينتهى المشروع بالكود

```
</mx:Application>
```

اما الاكواد الخاصة برسم مربعات النص و الزر و غيرها تكتب فى المنتصف بين الكودين السابقين .

لاحظ ان : كل تاج فى ملف mxml يبدأ بحرفى mx و الذى يعتبر namespace صمم خصيصا لبرنامج flex.

عندما تقوم بتشغيل البرنامج سوف تشاهد التالى



لاحظ ان مشروعك فى النهاية ليس سوى ملف فلاش بامتداد swf و تشاهدة من خلال مستعرض الويب الخاص بك و هنا نحن نستخدم برنامج Mozilla firefox .

MXML

ستلاحظ ان معظم اكواد MXML تتعلق بكلاسات AS 3.0 او خصائص لها حيث ان مترجم برنامج FLEX يرسل اكواد MXML على هيئة swf بايت يتكون منها فيلم الفلاش فى النهاية ليصبح swf .

سبق و اشرت لمثال على ال MXML و الان لنتناول

```
<mx:Button id="myButton" label="Click me!" />
```

تلاحظ ان الخاصية id للزر قد اعطيت للزر mybutton لكى يكون مميز عن اى كنترول او عنصر اخر فى النموذج و الخاصية label يكتب فيها الاسم الذى يظهر على الزر

مثال :

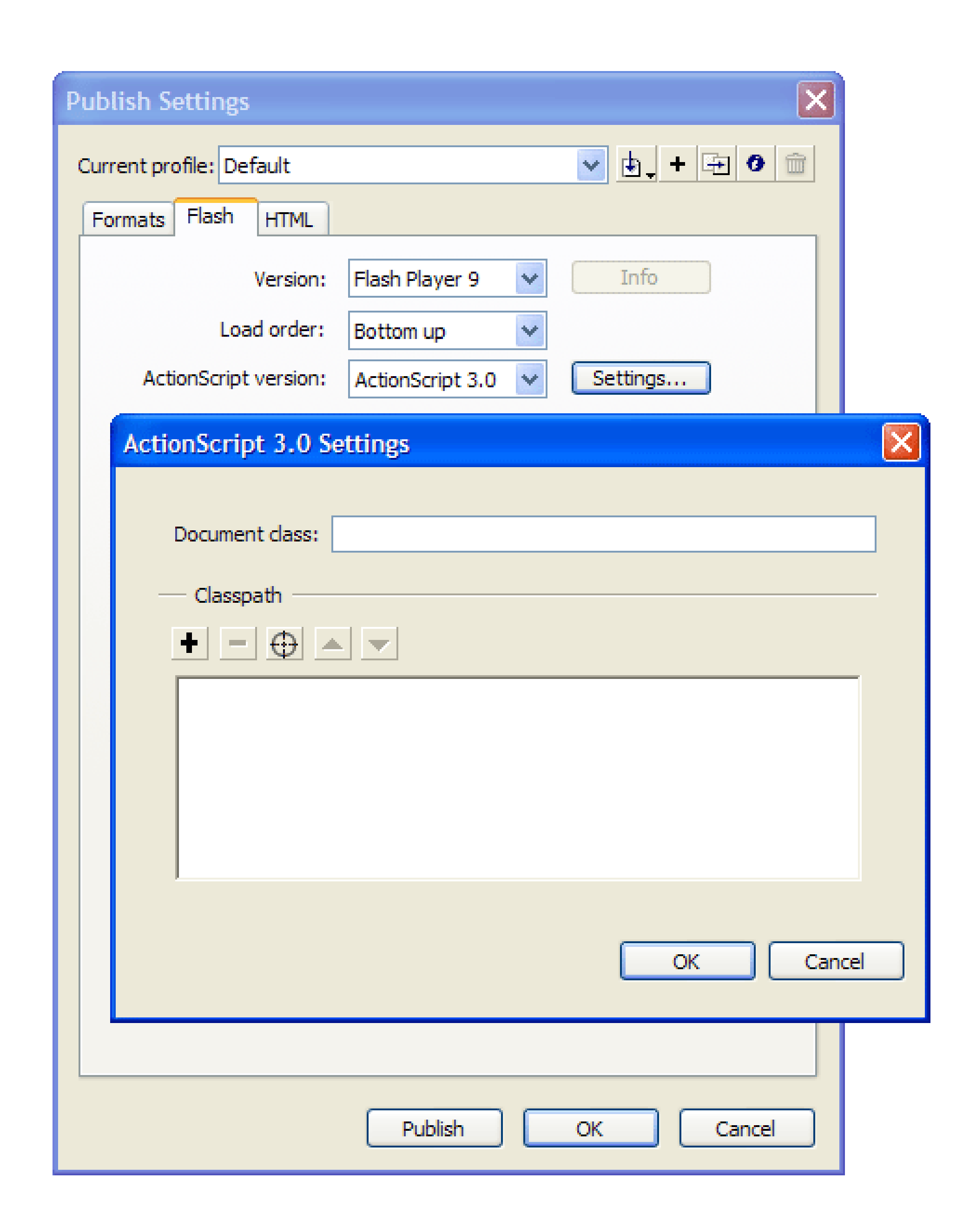
```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml"
    horizontalAlign="center" verticalAlign="center">

    <mx:Button id="myButton" label="Click me!" />

</mx:Application>
```

Document class: مفهوم جديد فى فلاش ٩ و معناة تعريف class فى الشريط الزمنى لملف الفلاش لكى يقوم ببناء الclass عند تشغيل ملف swf و يمكنك عمل ذلك من خلال النافذة Property inspector او من خلال مربع الحوار ActionScript 3.0 Publish .

- **File > Publish Settings > Flash tab > Settings button (انظر الشكل).**



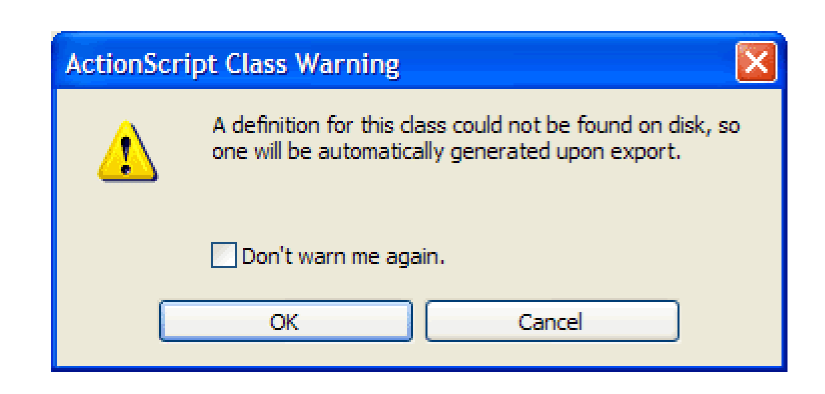
ActionScript 3.0 Settings dialog box

Symbol-class linkage: الان تستطيع ان تستخدم الترابط بين الclass لتقوم بتعريف instances من هذا الclass مباشرة.

Linkage Properties

Identifier:
Class: Symbol1
Linkage: Export for ActionScript
Export for runtime sharing
Export in first frame
Import for runtime sharing
URL:
OK
Cancel

ربما لا يستطيع الفلاش ايجاد الclass على الهارد ديسك لذلك سيقوم البرنامج بعمل class



مثال : عمل شكل يستجيب للنقر بالفأرة و للسحب .

١- قم بعمل ملف فلاش جديد و قم بتسميتة simpleBall.fla ثم قم برسم دائرة على مسرح العمل ثم قم بتحويلها الى موفى كليب واسمة circle و مربع Instance Name قم بأعطاء الموفى كليب اسم و ليكن ball_mc ثم قم بالنقر على الفريم الاول فى الشريط الزمنى و اضغط f9

٢- ثم اكتب الكود التالى :

```
ball_mc.addEventListener(MouseEvent.CLICK, clickHandler);
function clickHandler(event:MouseEvent):void {
  trace("You clicked the ball");
}
```

٣- لقد قمنا بعمل مستجيب للحدث و الحدث هنا هو النقر على الموفى كليب و استخدمنا MouseEvent.CLICK بدلا من onPress() التى كانت مستخدمة فى AS2 .

٤- الان لكى نعطى المستخدم الاحساس بأن الكرة عبارة عن زر نكتب الكود التالى :

```
ball_mc.buttonMode = true;
```

٥- قم بتجربة الملف control+enter .

٦- الان لكى نستطيع سحب و افلات الكرة

سوف نستخدم الاحداث MouseEvent.MOUSE_DOWN ،MouseEvent.MOUSE_UP

الكود:

```
ball_mc.buttonMode = true;
ball_mc.addEventListener(MouseEvent.CLICK, clickHandler);
ball_mc.addEventListener(MouseEvent.MOUSE_DOWN,
mouseDownListener);
```

```
ball_mc.addEventListener(MouseEvent.MOUSE_UP, mouseUpListener);
function clickHandler(event:MouseEvent):void {
  trace("You clicked the ball");
}
function mouseDownListener(event:MouseEvent):void {
  ball_mc.startDrag();
}
function mouseUpListener(event:MouseEvent):void {
  ball_mc.stopDrag();
}
```

الان قم بتجربة العمل ستجد انك تسطيع تحريك الكرة فى اى مكان على مسرح العمل و تركة اى اصبحت الكرة تتمتع بخاصية السحب و الافلات و لكن تصور انك تريد عمل اشكال كثيرة تتمتع بنفس الخاصية ولا يعقل ان نقوم بتكرار الكود مع كل شكل لكن من الاسهل عمل class تكون و ظيفتها توفير خاصية السحب و الافلات لكل شكل نقوم بعملة فى الفيلم بعد ذلك سنقوم بتحويل الكود السابق لclass .

٧- قم بعمل ملف اكشن اسكربت جديد من خلال

File > New > ActionScript File

ثم قم بحفظة فى نفس المجلد الموجود بة الملف السابق و ليكن اسم الملف Ball.as

ثم اكتب الكود التالى فى ملف الاكشن

```
package {
  import flash.display.MovieClip;
  import flash.events.MouseEvent;
  public class Ball extends MovieClip {
    public function Ball() {
      trace("ball created: " + this.name);
      this.buttonMode = true;
      this.addEventListener(MouseEvent.CLICK, clickHandler);
      this.addEventListener(MouseEvent.MOUSE_DOWN,
mouseDownListener);
      this.addEventListener(MouseEvent.MOUSE_UP, mouseUpListener);
    }
    private function clickHandler(event:MouseEvent):void {
      trace("You clicked the ball");
    }
    function mouseDownListener(event:MouseEvent):void {
      this.startDrag();
    }
    function mouseUpListener(event:MouseEvent):void {
      this.stopDrag();
    }
  }
}
```

٨- الان قم بالعودة لملف الفلاش السابق و ارسم كرة و حولها لموفى كليب ثم اذهب الى مكتبة الملفات و انقر بالزر الايمن للفأرة على الموفى كليب الجديد و اختر الامر Linkage من القائمة ثم اختر Export و فى Class text box اكتب Ball ثم اضغط على ok

٩- الان جرب العمل ستجد ان الدائرة الجديدة اصبحت لها خاصية السحب و الافلات .

الفصل السابع عشر

الاختلافات بين اكشن اسكريبت الاصدار ٢ و الاصدار الحديث ٣

سنجد ان هناك تعبيرات و مصطلاحات قد تم ازالتها من اللغة و ايضا هناك خصائص تم تعديلها او تغييرها لذلك الجدول التالى سيوضح الفرق بين الاصدارين :

ActionScript 2.0	ActionScript 3.0	Comments
Compiler directives		
#endinitclip	Removed	
#include	Removed	See the include statement for similar functionality.
#initclip	Removed	
Constants		
false	false	The value false, rather than undefined, is the default value of a Boolean object.
NaN	NaN	The value NaN, rather than undefined, is the default value of a Number object.
newline	Removed	Use the escape sequence composed of the backslash character followed by the character "n" (\n).
null	null	The value null, rather than undefined, is the default value of the Object and String classes.
undefined	undefined	The value undefined can be assigned only to untyped variables; it is not the default value of any typed object.

Global functions		
asfunction	flash.text.TextField dispatches event: link	Replaced by the new event handling model. You now get the same functionality by using the syntax Event: instead of asfunction:. When a user clicks the link, Flash Player dispatches a TextEvent object of type TextEvent.LINK, which your code can listen for with the addEventListener() method. Any text that you decide to include is stored in the event object's text property.
call()	Removed	
chr()	Removed	
clearInterval()	flash.utils.clearInterval()	Moved to flash.utils package.
clearTimeout()	flash.utils.clearTimeout()	Moved to flash.utils package.
duplicateMovieClip()	flash.display.MovieClip.MovieClip()	Replaced by new MovieClip class constructor function.
eval()	Removed	
fscommand()	flash.system.fscommand()	Moved to flash.system package. Also, see flash.external.ExternalInterface class for JavaScript/ActionScript communication.
getProperty()	Removed	To directly access properties, use the dot (.) operator.
getTimer()	flash.utils.getTimer()	Moved to flash.utils package.
getURL()	flash.net.navigateToURL()	Replaced by the navigateToURL() function.
getVersion()	flash.system.Capabilities.version	Moved to Capabilities

		class and changed to accessor property.
gotoAndPlay()	flash.display.MovieClip.gotoAndPlay()	This function is no longer a global function, but is still available as a method of the MovieClip class.
gotoAndStop()	flash.display.MovieClip.gotoAndStop()	This function is no longer a global function, but it is still available as a method of the MovieClip class.
ifFrameLoaded()	flash.display.MovieClip.framesLoaded	
int()	int()	Resurrected from deprecated status as a conversion function for the new int data type.
length()	String.length	This property is no longer a global property, but it is still available as a property of the String class.
loadMovie()	flash.display.Loader	Use the Loader class instead.
loadMovieNum()	flash.display.Loader	Use the Loader class instead.
loadVariables()	flash.net.URLLoader	Use the URLLoader class instead.
loadVariablesNum()	flash.net.URLLoader	Use the URLLoader class instead.
mbchr()	String.fromCharCode()	Removed. Use the static String.fromCharCode() method instead.
mblength()	String.length	Removed. Use String.length instead.
mbord()	String.charCodeAt()	Removed. Use String.charCodeAt() instead.
mbsubstring()	String.substr()	Removed. Use String.substr() instead.
nextFrame()	flash.display.MovieClip.nextFrame(	This function is no

	)	longer a global function, but it is still available as a method of the MovieClip class.
nextScene()	flash.display.MovieClip.nextScene()	This function is no longer a global function, but it is still available as a method of the MovieClip class.
on()	flash.events.EventDispatcher	Removed. Use the new event handling system in the flash.events package instead.
onClipEvent()	flash.events.EventDispatcher	Removed. Use the new event handling system in the flash.events package instead.
ord()	String	Removed. Use String class methods instead.
parseInt()	parseInt()	A string with a leading 0 is interpreted as decimal rather than octal. For octal numbers, pass the number 8 for the radix parameter.
play()	flash.display.MovieClip.play()	This function is no longer a global function, but it is still available as a method of the MovieClip class.
prevFrame()	flash.display.MovieClip.prevFrame()	This function is no longer a global function, but it is still available as a method of the MovieClip class.
prevScene()	flash.display.MovieClip.prevScene()	This function is no longer a global function, but it is still available as a method of the MovieClip class.
print()	flash.printing.PrintJob	Removed. Use the PrintJob class instead.

printAsBitmap()	flash.printing.PrintJob	Removed. Use the PrintJob class instead.
printAsBitmapNum()	flash.printing.PrintJob	Removed. Use the PrintJob class instead.
printNum()	flash.printing.PrintJob	Removed. Use the PrintJob class instead.
random()	Math.random()	Removed. Use Math.random() instead.
removeMovieClip()	Removed	Set all references to a movie clip to null to make the movie clip eligible for garbage collection.
setInterval()	flash.utils.setInterval()	Moved to flash.utils package. Consider using the Timer class instead.
setProperty()	Removed	To set the value of a writable property, use the dot (.) operator.
setTimeout()	flash.utils.setTimeout()	Moved to flash.utils package.
showRedrawRegions( )	flash.profiler.showRedrawRegions()	Moved to flash.profiler package.
startDrag()	flash.display.Sprite.startDrag()	This is no longer a global function, but it is still available as a method of the Sprite class.
stop()	flash.display.MovieClip.stop()	This is no longer a global function, but it is still available as a method of the MovieClip class.
stopAllSounds()	flash.media.SoundMixer.stopAll()	This is no longer a global function, but it is still available as a method of the SoundMixer class, which provides global sound control.
stopDrag()	flash.display.Sprite.stopDrag()	This is no longer a global function, but it is still available as a

		method of the Sprite class.
substring()	String.substring()	This is no longer a global function, but it is still available as a method of the String class.
targetPath()	Removed	
tellTarget()	Removed	Use the dot (.) operator or the with statement instead.
toggleHighQuality()	flash.display.Stage.quality	Removed as a global property. Use the Stage class version instead.
trace()	trace()	The trace() method accepts a comma-delimited list of arguments.
unloadMovie()	flash.display.Loader.unload()	Removed. Use Loader.unload() instead.
unloadMovieNum()	flash.display.Loader.unload()	Removed. Use Loader.unload() instead.
updateAfterEvent()	flash.events.TimerEvent.updateAfterEvent()	This is no longer a global function, but it is still available as a method of the TimerEvent, MouseEvent, and KeyboardEvent classes.
Global properties		
_accProps	flash.accessibility.AccessibilityProperties	Replaced by the AccessibilityProperties class.
_focusrect	flash.display.InteractiveObject.focusRect	Replaced by the focusRect property of the InteractiveObject class.
_global	Removed	Use a static member of a class instead.

_highquality	flash.display.Stage.quality	Replaced by the quality property of the Stage class.
_level	Removed	The concept of levels does not exist in ActionScript 3.0, which instead provides direct access to the display list. See the flash.display package for details.
maxscroll	flash.text.TextField	Replaced by the maxScrollH and maxScrollV properties of the TextField class.
_parent	flash.display.DisplayObject.parent	Replaced by the parent property of the DisplayObject class.
_quality	flash.display.Stage.quality	Replaced by the quality property of the Stage class.
_root	flash.display.DisplayObject.stage	Removed. The closest equivalent is the Stage, which serves as the root of the ActionScript 3.0 display list.
scroll	flash.text.TextField	Removed. Replaced by the scrollH and scrollV properties of the TextField class.
_soundbuftime	flash.media.SoundMixer.bufferTime	Replaced by the bufferTime property of the SoundMixer class.
this	this	Instance methods are bound to the instance that implemented the method; therefore, the this reference inside the body of an instance method always refers to the instance that implemented the method.

Operators		
add (concatenation (strings))	Removed	Use the concatenation (+) operator instead.
eq (equality (strings))	Removed	Use the equality (==) operator instead.
gt (greater than (strings))	Removed	Use the greater than (>) operator instead.
ge (greater than or equal to (strings))	Removed	Use the greater than or equal to (>=) operator instead.
<> (inequality)	Removed	Use the inequality (!=) operator instead.
instanceof	is	Although the instanceof operator is available, it only checks the prototype chain, which is not the primary inheritance mechanism in ActionScript 3.0. Use the is operator to check whether an object is a member of a specific data type.
lt (less than (strings))	Removed	Use the less than (<) operator instead.
le (less than or equal to (strings))	Removed	Use the less than or equal to (<=) operator instead.
and (logical AND)	Removed	Use the logical AND (&&) operator instead.
not (logical NOT)	Removed	Use the logical NOT (!) operator instead.
or (logical OR)	Removed	Use the logical OR (||) operator instead.
ne (not equal (strings))	Removed	Use the inequality (!=) operator instead.
Statements		
delete	delete	The delete operator works only on properties of objects, but not on variables

		that hold references to objects.
import	import	The import statement is not optional. To use a class, you must import it, whether or not you use a fully qualified name.
intrinsic	Removed	ActionScript 3.0 has a similar, but not identical, keyword named native. The native keyword is similar to intrinsic in that it directs the compiler not to expect a function body, but is dissimilar in that native has no effect on compile-time type checking.
private	private	The ActionScript 3.0 private keyword specifies that an identifier is visible only within a class, and does not extend to subclasses. Moreover, in ActionScript 3.0 the private keyword is enforced at both compile time and run time.
set variable	Removed	Use the assignment (=) operator instead.
super	super	In ActionScript 3.0, the call to super() in a subclass constructor does not have to be the first statement in the constructor body.

Accessibility class	flash.accessibility.Accessibility	
isActive() Method	flash.accessibility.Accessibility.active	Changed from function to accessor property.

		Name changed from isActive to active.
updateProperties() Method	flash.accessibility.Accessibility.updateProperties()	
arguments class	arguments	
caller Property	Removed	You can achieve the same functionality by passing arguments.callee from the caller function as an argument to the callee function. See the Examples section of arguments.callee for an example.
Array class		
CASEINSENSITIVE Constant	Array.CASEINSENSITIVE	Data type changed to uint.
DESCENDING Constant	Array.DESCENDING	Data type changed to uint.
length Property	Array.length	Data type changed to uint.
NUMERIC Constant	Array.NUMERIC	Data type changed to uint.
RETURNINDEXEDARRAY Constant	Array.RETURNINDEXEDARRAY	Data type changed to uint.
UNIQUESORT Constant	Array.UNIQUESORT	Data type changed to uint.
Array Constructor	Array.Array()	Parameter changed to use the ...(rest) parameter format.
push() Method	Array.push()	Parameter changed to use the ...(rest) parameter format.
sort() Method	Array.sort()	Data type of the options parameter changed to uint.
sortOn() Method	Array.sortOn()	Data type of the options parameter changed to uint. The ActionScript 3.0

		version also has added functionality; you can now sort on more than one field name by passing an array of objects for the fieldName parameter, and each sort field can have its own matching options parameter if you also pass in a matching array of options flags for the options parameter.
splice() Method	Array.splice()	The parameters can have any data type, but the preferred data types are int and uint. The value parameter changed to the ...(rest) parameter format.
unshift() Method	Array.unshift()	The value parameter changed to the ...(rest) format. Data type of the return value changed to uint.
AsBroadcaster class	flash.events.EventDispatcher	
_listeners Property[read-only]	flash.events.EventDispatcher.willTrigger()	Not a direct equivalent. The willTrigger() method tells you whether any listeners are registered, but not how many.
addListener() Method	flash.events.EventDispatcher.addEventListener()	Not a direct equivalent, because the ActionScript 3.0 event model lets you add event listeners to any object in the event flow, not just to the broadcasting object.
broadcastMessage() Method	flash.events.EventDispatcher.dispatchEvent()	Not a direct equivalent, because the ActionScript 3.0 event

		model works differently. The dispatchEvent() method dispatches an event object into the event flow, while the broadcastMessage() method sends messages directly to each registered listener object.
initialize() Method	Removed	There is no direct equivalent in ActionScript 3.0, but similar functionality is achieved by subclassing the EventDispatcher class. For example, the DisplayObject class extends EventDispatcher, so all instances of the DisplayObject and DisplayObject subclasses are capable of sending and receiving event objects.
removeListener() Method	flash.events.EventDispatcher.removeEventListener()	Not a direct equivalent, because the ActionScript 3.0 event model lets you add event listeners to and remove them from any object in the event flow, not just the broadcasting object.
BitmapData class	flash.display.BitmapData	ActionScript 3.0 uses the BitmapDataChannel class as an enumeration of constants that indicate which channel to use.

height Property[read-only]	flash.display.BitmapData.height	Data type changed from Number to int.
rectangle Property[read-only]	flash.display.BitmapData.rect	Property renamed for consistency with other members of the API.
width Property[read-only]	flash.display.BitmapData.width	Data type changed from Number to int.
copyChannel() Method	flash.display.BitmapData.copyChannel()	The sourceChannel and destChannel parameters are now uint data types.
draw() Method	flash.display.BitmapData.draw()	The source parameter is now IBitmapDrawable; DisplayObject and BitmapData both implement the IBitmapDrawable interface, so you can pass either a DisplayObject or a BitmapData object to the source parameter.
fillRect() Method	flash.display.BitmapData.fillRect()	The color parameter is now a uint value.
floodFill() Method	flash.display.BitmapData.floodFill()	Now accepts int values for the x and y parameters and a uint value for color.
getColorBoundsRect() Method	flash.display.BitmapData.getColorBoundsRect()	Now accepts uint values for the mask and color parameters.
getPixel() Method	flash.display.BitmapData.getPixel()	Now accepts int parameter values and returns a uint value.
getPixel32() Method	flash.display.BitmapData.getPixel32()	Now accepts int parameter values and returns a uint value.
hitTest() Method	flash.display.BitmapData.hitTest()	Now accepts uint values for the firstAlphaThreshold and secondAlphaThreshold parameters.

loadBitmap() Method	Removed	
merge() Method	flash.display.BitmapData.merge()	Now accepts uint values for the multiplier parameters.
noise() Method	flash.display.BitmapData.noise()	Now accepts an int value for the randomSeed parameter and uint values for the low, high, and channelOptions parameters.
perlinNoise() Method	flash.display.BitmapData.perlinNoise()	Now accepts an int value for the randomSeed parameter and uint values for the numOctaves and channelOptions parameters.
pixelDissolve() Method	flash.display.BitmapData.pixelDissolve()	Now accepts an int value for the randomSeed and numPixels parameters and a uint value for the fillColor parameter. (The numPixels parameter is named numberOfPixels in ActionScript 2.0.)
scroll() Method	flash.display.BitmapData.scroll()	Now accepts int values for the x and y parameters.
setPixel() Method	flash.display.BitmapData.setPixel()	Now accepts int values for the x and y parameters and a uint value for color.
setPixel32() Method	flash.display.BitmapData.setPixel32()	Now accepts int values for the x and y parameters and a unit value for color.
threshold() Method	flash.display.BitmapData.threshold()	Now accepts uint values for the threshold, color, and mask parameters, and

		returns a uint value.
BlurFilter class		
quality Property	flash.filters.BlurFilter.quality	The quality property data type changed from a Number to uint.
Button class	flash.display.SimpleButton	
_alpha Property	flash.display.DisplayObject.alpha	
blendMode Property	flash.display.DisplayObject.blendMode	
cacheAsBitmap Property	flash.display.DisplayObject.cacheAsBitmap	
enabled Property	flash.display.SimpleButton.enabled	
filters Property	flash.display.DisplayObject.filters	In ActionScript 3.0, the data type is Array.
_focusrect Property	flash.display.InteractiveObject.focusRect	
_height Property	flash.display.DisplayObject.height	
_highquality Property	Removed	
_name Property	flash.display.DisplayObject.name	
_parent Property	flash.display.DisplayObject.parent	
_quality Property	Removed	You can set rendering quality for all display objects by using flash.display.Stage.quality.
_rotation Property	flash.display.DisplayObject.rotation	
scale9Grid Property	flash.display.DisplayObject.scale9Grid	
_soundbuftime Property	flash.media.SoundMixer.bufferTime	Moved to the SoundMixer class, which is used for global sound control. Renamed without abbreviations. Removed the initial underscore from the name.
tabEnabled Property	flash.display.InteractiveObject.tabEnabled	

tabIndex Property	flash.display.InteractiveObject.tabIndex	
_target Property[read-only]	Removed	
trackAsMenu Property	flash.display.SimpleButton.trackAsMenu	
_url Property[read-only]	Removed	
useHandCursor Property	flash.display.SimpleButton.useHandCursor	
_visible Property	flash.display.DisplayObject.visible	
_width Property	flash.display.DisplayObject.width	
_x Property	flash.display.DisplayObject.x	
_xmouse Property[read-only]	flash.display.DisplayObject.mouseX	
_xscale Property	flash.display.DisplayObject.scaleX	
_y Property	flash.display.DisplayObject.y	
_ymouse Property[read-only]	flash.display.DisplayObject.mouseY	
_yscale Property	flash.display.DisplayObject.scaleY	
getDepth() Method	flash.display.DisplayObjectContainer.getChildIndex()	ActionScript 3.0 provides direct access to the display list, so depth is handled differently.
onDragOut() EventHandler	flash.display.InteractiveObject dispatches event: mouseOut	Replaced in the new event model by a mouseOut event, after a call to InteractiveObject.setCapture().
onDragOver() EventHandler	flash.display.InteractiveObject dispatches event: mouseOver	Replaced in the new event model by a mouseOver event after a call to the InteractiveObject.setCapture() method.
onKeyDown() EventHandler	flash.display.InteractiveObject dispatches event: keyDown	Replaced in the new event model by a keyDown event.
onKeyUp() EventHandler	flash.display.InteractiveObject dispatches event: keyUp	Replaced in the new event model by a keyUp event.

onKillFocus() EventHandler	flash.display.InteractiveObject dispatches event: focusOut	Replaced in the new event model by a focusOut event.
onPress() EventHandler	flash.display.InteractiveObject dispatches event: mouseDown	Replaced in the new event model by a mouseDown event.
onRelease() EventHandler	flash.display.InteractiveObject dispatches event: mouseUp	Replaced in the new event model by a mouseUp event.
onReleaseOutside() EventHandler	flash.display.InteractiveObject dispatches event: mouseUp	Replaced in the new event model by a mouseUp event after a call to flash.display.InteractiveObject.setCapture().
onRollOut() EventHandler	flash.display.InteractiveObject dispatches event: mouseOut	Replaced in the new event model by a mouseOut event.
onRollOver() EventHandler	flash.display.InteractiveObject dispatches event: mouseOver	Replaced in the new event model by a mouseOver event.
onSetFocus() EventHandler	flash.display.InteractiveObject dispatches event: focusIn	Replaced in the new event model by a focusIn event.

Camera class	flash.media.Camera	
activityLevel Property[read-only]	flash.media.Camera.activityLevel	
bandwidth Property[read-only]	flash.media.Camera.bandwidth	
currentFps Property[read-only]	flash.media.Camera.currentFPS	Change in capitalization of FPS.
fps Property[read-only]	flash.media.Camera.fps	
height Property[read-only]	flash.media.Camera.height	Data type changed from Number to int.
index Property[read-only]	flash.media.Camera.index	Data type changed from String to int.
motionLevel Property[read-only]	flash.media.Camera.motionLevel	Data type changed from Number to int.
motionTimeOut Property[read-only]	flash.media.Camera.motionTimeout	Data type changed from Number to int.
muted Property[read-	flash.media.Camera.muted	

only]		
name Property[read-only]	flash.media.Camera.name	
names Property[read-only]	flash.media.Camera.names	
quality Property[read-only]	flash.media.Camera.quality	Data type changed from Number to int.
width Property[read-only]	flash.media.Camera.width	Data type changed from Number to int.
get() Method	flash.media.Camera.getCamera()	
onActivity() EventHandler	flash.events.ActivityEvent.ACTIVITY	
onStatus() EventHandler	flash.media.Camera dispatches event: status	Replaced in the new event model by a status StatusEvent object.
setMode() Method	flash.media.Camera.setMode()	The width and height parameters changed to data type int.
setMotionLevel() Method	flash.media.Camera.setMotionLevel()	Both parameters changed to data type int.
setQuality() Method	flash.media.Camera.setQuality()	Both parameters changed to data type int.

capabilities class	flash.system.Capabilities	The class name changed from lowercase to initial capitalization.

Color class	flash.geom.ColorTransform	The Color class has been removed because all of its functionality can be achieved with the flash.geom.ColorTransform class. Color values can be assigned directly by using the ColorTransform class constructor or

		properties. ColorTransform objects can then be assigned to the colorTransform property of a Transform object, which in turn can be assigned to the transform property of a DisplayObject instance.
Color Constructor	flash.geom.ColorTransform.ColorTransform()	Removed. You can specify color values by using the ColorTransform() constructor.
getRGB() Method	flash.geom.ColorTransform.color	The RGB color value can be accessed by using the color accessor property of the ColorTransform class.
getTransform() Method	Removed	Color values can be assigned directly by using the ColorTransform() class constructor or properties.
setRGB() Method	flash.geom.ColorTransform.color	The RGB color value can be set by using the color accessor property of the ColorTransform class.
setTransform() Method	Removed	Color values can be assigned directly by using the ColorTransform() class constructor or properties.
ContextMenu class	flash.ui.ContextMenu	The ContextMenu class is now part of the flash.ui package.
builtInItems Property	flash.ui.ContextMenu.builtInItems	

customItems Property	flash.ui.ContextMenu.customItems	
ContextMenu Constructor	flash.ui.ContextMenu.ContextMenu()	
copy() Method	flash.ui.ContextMenu.clone()	
hideBuiltInItems() Method	flash.ui.ContextMenu.hideBuiltInItems()	
onSelect() EventHandler	flash.ui.ContextMenu dispatches event: menuSelect	Instead of invoking the onSelect() event handler, the ActionScript 3.0 class dispatches a menuSelect event.
ContextMenuItem class	flash.ui.ContextMenuItem	The ContextMenuItem class is now part of the flash.ui package.
caption Property	flash.ui.ContextMenuItem.caption	
enabled Property	flash.ui.ContextMenuItem.enabled	
separatorBefore Property	flash.ui.ContextMenuItem.separatorBefore	
visible Property	flash.ui.ContextMenuItem.visible	
ContextMenuItem Constructor	flash.ui.ContextMenuItem.ContextMenuItem()	
copy() Method	flash.ui.ContextMenuItem.clone()	
onSelect() EventHandler	flash.ui.ContextMenuItem dispatches event: menuItemSelect	Instead of invoking the onSelect() event handler, the ActionScript 3.0 class dispatches a menuSelect event.
ConvolutionFilter class		
clone() Method	flash.filters.ConvolutionFilter.clone()	Now returns a BitmapFilter object.
Date class	Date	ActionScript 3.0 includes a new set of read accessors for all the methods that start with get*xxx*(). For example, in

		ActionScript 3.0, Date.getDate() and Date.date return the same value.
getUTCYear() Method	Date.getUTCFullYear()	This method was removed because it is not part of ECMAScript. Use Date.getUTCFullYear() instead.
getYear() Method	Date.getFullYear()	This method was removed because it is not part of ECMAScript. Use Date.getFullYear() instead.
setYear() Method	Date.setFullYear()	This method was removed because it is not part of ECMAScript. Use Date.setFullYear() instead.

DisplacementMapFilter class	flash.filters.DisplacementMapFilter	The data type of several parameters changed from Number to uint.
color Property	flash.filters.DisplacementMapFilter.color	The data type of this parameter is now uint.
componentX Property	flash.filters.DisplacementMapFilter.componentX	The data type of this parameter is now uint.
componentY Property	flash.filters.DisplacementMapFilter.componentY	The data type of this parameter is now uint.
DisplacementMapFilter Constructor	flash.filters.DisplacementMapFilter.DisplacementMapFilter()	The data type of the componentX, componentY, and color parameters is now uint.
clone() Method	flash.filters.DisplacementMapFilter.clone()	Now returns a BitmapFilter object.

DropShadowFilter class	flash.filters.DropShadowFilter	
color Property	flash.filters.DropShadowFilter.color	The data type of this parameter changed

		from Number to uint.
quality Property	flash.filters.DropShadowFilter.quality	The data type of this parameter changed from Number to uint.
DropShadowFilter Constructor	flash.filters.DropShadowFilter.DropShadowFilter()	All parameters now have a default value, and some parameter types have changed.
clone() Method	flash.filters.DropShadowFilter.clone()	Now returns a BitmapFilter object instead of a DropShadowFilter object.
Error class	Error	Added a new getStackTrace() method to assist in debugging.
ExternalInterface class	flash.external.ExternalInterface	Parameters changed for two methods in this class.
addCallback() Method	flash.external.ExternalInterface.addCallback()	The ActionScript 3.0 version of this method does not accept the instance parameter. The method parameter is replaced by a closure parameter, which can take a reference to a function, a class method, or a method of a particular class instance. In addition, if the calling code cannot access the closure reference for security reasons, a SecurityError exception is thrown.
call() Method	flash.external.ExternalInterface.call()	If a problem occurs, the ActionScript 3.0 version of this method throws an Error or SecurityError

		exception, in addition to returning null.
FileReference class	flash.net.FileReference	The ActionScript 3.0 version inherits the addEventListener() and removeEventListener() methods from the EventDispatcher class. Dispatched events replace the event handler functions.
postData Property	flash.net.URLRequest.data	The postData property is added to ActionScript 2.0 in Flash Player 9 to send POST data with the file upload or download. In ActionScript 3.0, use the data property of the URLRequest class to send either POST or GET data. See flash.net.URLRequest.data in this language reference for details.
size Property[read-only]	flash.net.FileReference.size	Returns a uint data type instead of a Number data type.
addListener() Method	flash.events.EventDispatcher.addEventListener()	In the new event model, there is no need to have a class-specific addListener() method, because the class inherits the addEventListener() method from the EventDispatcher class.
browse() Method	flash.net.FileReference.browse()	In ActionScript 2.0, returns false when there is an error. In ActionScript 3.0, throws an IllegalOperationError or ArgumentError

		exception. However, the method still returns false if the parameters are invalid, the file-browsing dialog box does not open, or another browser session is in progress. Also, the typelist parameter changed. In ActionScript 2.0, you can pass the browse() method an array of strings to specify a file filter. In ActionScript 3.0, you pass an array of FileFilter objects.
download() Method	[flash.net.FileReference.download()]()	When an error occurs, throws exceptions instead of returning false. The data type for the first parameter has changed. In ActionScript 2.0, the first parameter you pass to download() is a string. In ActionScript 3.0, you pass a URLRequest object.
removeListener() Method	[flash.events.EventDispatcher.removeEventListener()]()	In the new event model, there is no need to have a class-specific removeListener() method, because the class inherits the removeEventListener() method from the EventDispatcher class.
upload() Method	[flash.net.FileReference.upload()]()	Various changes have occurred: • The data type for the first parameter has changed. In ActionScript 2.0,

		the first parameter you pass to upload() is a string. In ActionScript 3.0, you pass a URLRequest object. • In ActionScript 3.0, there is a new second parameter, uploadDataFieldName, which is the field name that precedes the file data in the upload POST operation. • In ActionScript 3.0, there is a new third parameter, testUpload, that lets you control whether Flash Player performs a test upload before uploading the file. • When an error occurs, browse() throws exceptions instead of returning false.
onCancel Listener	flash.net.FileReference dispatches event: cancel	In ActionScript 3.0, instead of invoking the onCancel() event handler, this class dispatches an event named cancel.
onComplete Listener	flash.net.FileReference dispatches event: complete	In ActionScript 3.0, instead of invoking the onComplete() event handler, this class

		dispatches an event named complete.
onHTTPError Listener	flash.net.FileReference dispatches event: httpStatus	In ActionScript 3.0, instead of invoking the onHTTPError() event handler, this class dispatches an event named httpStatus.
onIOError Listener	flash.net.FileReference dispatches event: ioError	In ActionScript 3.0, instead of invoking the onIOError() event handler, this class dispatches an event named ioError.
onOpen Listener	flash.net.FileReference dispatches event: open	In ActionScript 3.0, instead of invoking the onOpen() event handler, this class dispatches an event named open.
onProgress Listener	flash.net.FileReference dispatches event: progress	In ActionScript 3.0, instead of invoking the onProgress() event handler, this class dispatches an event named progress.
onSecurityError Listener	flash.net.FileReference dispatches event: securityError	In ActionScript 3.0, instead of invoking the onSecurityError() event handler, this class dispatches an event named securityError.
onSelect Listener	flash.net.FileReference dispatches event: select	In ActionScript 3.0, instead of invoking the onSelect() event handler, this class dispatches an event named select.
FileReferenceList class	flash.net.FileReferenceList	The ActionScript 3.0 class inherits the addEventListener() and removeEventListener() methods from the

		EventDispatcher class. Instead of the onCancel() and onSelect() event handlers, the ActionScript 3.0 class uses events named cancel and select.
addListener() Method	flash.events.EventDispatcher.addEventListener()	In the new event model, there is no need to have a class-specific addListener() method, because the class inherits the addEventListener() method from the EventDispatcher class.
browse() Method	flash.net.FileReferenceList.browse()	In ActionScript 3.0, instead of returning false when there is an error, this method throws an IllegalOperationError exception. Also, the typelist parameter changed. In ActionScript 2.0, you can pass the browse() method an array of strings to specify a file filter. In ActionScript 3.0, you pass an array of FileFilter objects.
removeListener() Method	flash.events.EventDispatcher.removeEventListener()	In the new event model, there is no need to have a class-specific removeListener() method, because the class inherits the removeEventListener() method from the EventDispatcher class.
onCancel Listener	flash.net.FileReferenceList dispatches event: cancel	In ActionScript 3.0, instead of invoking the onCancel() event

		handler, this class dispatches an event named cancel.
onSelect Listener	flash.net.FileReferenceList dispatches event: select	In ActionScript 3.0, instead of invoking the onSelect() event handler, this class dispatches an event named select.
GlowFilter class	flash.filters.GlowFilter	The data type of several properties changed from Number to unit.
color Property	flash.filters.GlowFilter.color	The data type of this property changed from Number to unit.
quality Property	flash.filters.GlowFilter.quality	The data type of this property changed from Number to unit.
GlowFilter Constructor	flash.filters.GlowFilter.GlowFilter()	The color and quality parameters are now uint and int data types, respectively, instead of Number. All parameters are now assigned a default value.
clone() Method	flash.filters.GlowFilter.clone()	Returns a BitmapFilter object instead of a GlowFilter object.
GradientBevelFilter class	flash.filters.GradientBevelFilter	
quality Property	flash.filters.GradientBevelFilter.quality	The data type of this property changed from Number to int.
clone() Method	flash.filters.GradientBevelFilter.clone()	Returns a BitmapFilter object instead of a GradientBevelFilter object.
GradientGlowFilter class	flash.filters.GradientGlowFilter	

quality Property	flash.filters.GradientGlowFilter.quality	The data type of this property changed from Number to int.
GradientGlowFilter Constructor	flash.filters.GradientGlowFilter.GradientGlowFilter()	Default values added to all parameters and the data type of the quality parameter changed from Number to int.
clone() Method	flash.filters.GradientGlowFilter.clone()	Returns a BitmapFilter object instead of a GradientGlowFilter object.
IME class	flash.system.IME	This class has been moved to the flash.system package.
ALPHANUMERIC_FULL Constant	flash.system.IMEConversionMode.ALPHANUMERIC_FULL	
ALPHANUMERIC_HALF Constant	flash.system.IMEConversionMode.ALPHANUMERIC_HALF	
CHINESE Constant	flash.system.IMEConversionMode.CHINESE	
JAPANESE_HIRAGANA Constant	flash.system.IMEConversionMode.JAPANESE_HIRAGANA	
JAPANESE_KATAKANA_FULL Constant	flash.system.IMEConversionMode.JAPANESE_KATAKANA_FULL	
JAPANESE_KATAKANA_HALF Constant	flash.system.IMEConversionMode.JAPANESE_KATAKANA_HALF	
KOREAN Constant	flash.system.IMEConversionMode.KOREAN	
UNKNOWN Constant	flash.system.IMEConversionMode.UNKNOWN	
addListener() Method	flash.events.EventDispatcher.addEventListener()	In the new event model, there is no need to have a class-specific addListener() method, because the class inherits the addEventListener() method from the EventDispatcher class.
getConversionMode()	flash.system.IME.conversionMode	Changed to an

Method		accessor property.
getEnabled() Method	flash.system.IME.enabled	Changed to an accessor property.
removeListener() Method	flash.events.EventDispatcher.removeEventListener()	In the new event model, there is no need to have a class-specific removeListener() method, because the class inherits the removeEventListener() method from the EventDispatcher class.
setConversionMode() Method	flash.system.IME.conversionMode	Changed to an accessor property.
setEnabled() Method	flash.system.IME.enabled	Changed to an accessor property.
onIMEComposition Listener	flash.system.IME dispatches event: imeComposition	In ActionScript 3.0, instead of invoking the onIMEComposition() event handler, this class dispatches an event named imeComposition.
Key class	flash.ui.Keyboard	This class has a new name in ActionScript 3.0 to match other classes that pertain to the Keyboard class, such as KeyboardEvent.
BACKSPACE Constant	flash.ui.Keyboard.BACKSPACE	Declared as a constant in ActionScript 3.0 and data type changed to unit.
CAPSLOCK Constant	flash.ui.Keyboard.CAPS_LOCK	Declared as a constant in ActionScript 3.0, underscore added, and data type changed to uint.
CONTROL Constant	flash.ui.Keyboard.CONTROL	Declared as a constant in ActionScript 3.0 and data type changed to uint.

DELETEKEY Constant	flash.ui.Keyboard.DELETE	Name changed to DELETE in ActionScript 3.0, declared as a constant, and data type changed to uint.
DOWN Constant	flash.ui.Keyboard.DOWN	Declared as a constant in ActionScript 3.0 and data type changed to uint.
END Constant	flash.ui.Keyboard.END	Declared as a constant in ActionScript 3.0 and data type changed to uint.
ENTER Constant	flash.ui.Keyboard.ENTER	Declared as a constant in ActionScript 3.0 and data type changed to uint.
ESCAPE Constant	flash.ui.Keyboard.ESCAPE	Declared as a constant in ActionScript 3.0 and data type changed to uint.
HOME Constant	flash.ui.Keyboard.HOME	Declared as a constant in ActionScript 3.0 and data type changed to uint.
INSERT Constant	flash.ui.Keyboard.INSERT	Declared as a constant in ActionScript 3.0 and data type changed to uint.
LEFT Constant	flash.ui.Keyboard.LEFT	Declared as a constant in ActionScript 3.0 and data type changed to uint.
_listeners Property[read-only]	flash.events.EventDispatcher.willTrigger()	Not a direct equivalent. The willTrigger() method tells you whether any listeners are registered, but not how many.
PGDN Constant	flash.ui.Keyboard.PAGE_DOWN	Name changed to PAGE_DOWN in ActionScript 3.0, declared as a constant, and data type changed

		to uint.
PGUP Constant	flash.ui.Keyboard.PAGE_UP	Name changed to PAGE_UP in ActionScript 3.0, declared as a constant, and data type changed to uint.
RIGHT Constant	flash.ui.Keyboard.RIGHT	Declared as a constant in ActionScript 3.0 and data type changed to uint.
SHIFT Constant	flash.ui.Keyboard.SHIFT	Declared as a constant in ActionScript 3.0 and data type changed to uint.
SPACE Constant	flash.ui.Keyboard.SPACE	Declared as a constant in ActionScript 3.0 and data type changed to uint.
TAB Constant	flash.ui.Keyboard.TAB	Declared as a constant in ActionScript 3.0 and data type changed to uint.
UP Constant	flash.ui.Keyboard.UP	Declared as a constant in ActionScript 3.0 and data type changed to uint.
addListener() Method	flash.events.EventDispatcher.addEventListener()	In ActionScript 3.0, there is no need to have a class-specific addListener() method, because all display objects inherit the addEventListener() method from the EventDispatcher class.
getAscii() Method	flash.events.KeyboardEvent.charCode	
getCode() Method	flash.events.KeyboardEvent.keyCode	
isAccessible() Method	flash.ui.Keyboard.isAccessible()	
isDown() Method	Removed	Removed for security reasons.
isToggled() Method	Removed	Removed for security

		reasons.
removeListener() Method	flash.events.EventDispatcher.removeEventListener()	In ActionScript 3.0, there is no need to have a class-specific removeListener() method, because all display objects inherit the removeEventListener() method from the EventDispatcher class.
onKeyDown Listener	flash.display.InteractiveObject dispatches event: keyDown	In ActionScript 3.0, instead of invoking the onKeyDown event handler, the InteractiveObject class dispatches a keyDown KeyboardEvent object.
onKeyUp Listener	flash.display.InteractiveObject dispatches event: keyUp	In ActionScript 3.0, instead of invoking the onKeyUp event handler, the InteractiveObject class dispatches a keyUp KeyboardEvent object.
LoadVars class	flash.net.URLLoader	The LoadVars class functionality is replaced by the URLLoader, URLRequest, URLStream, and URLVariables classes.
contentType Property	flash.net.URLRequest.contentType	
loaded Property	Removed	There is no corresponding Boolean property in ActionScript 3.0, but you can use flash.events.Event.COMPLETE to set up listeners that receive notification when data is loaded.
LoadVars Constructor	flash.net.URLLoader.URLLoader()	
addRequestHeader()	flash.net.URLRequestHeader	

Method		
decode() Method	flash.net.URLVariables.decode()	
getBytesLoaded() Method	flash.net.URLLoader.bytesLoaded	Class changed to URLLoader; changed from function to property accessor; and name changed from getBytesLoaded to bytesLoaded.
getBytesTotal() Method	flash.net.URLLoader.bytesTotal	Class changed to URLLoader; changed from function to property accessor; and name changed from getBytesTotal to bytesTotal.
load() Method	flash.net.URLLoader.load()	
onData() EventHandler	flash.net.URLLoader dispatches event: complete	See the URLLoader class. A complete event is dispatched when the download operation is complete but before any data is parsed.
onHTTPStatus() EventHandler	flash.net.URLLoader dispatches event: httpStatus	In ActionScript 3.0, instead of invoking the onHTTPStatus event handler, the URLLoader class dispatches an HTTPStatusEvent object named httpStatus.
onLoad() EventHandler	flash.net.URLLoader dispatches event: complete	See the URLLoader class. A complete event is dispatched when the download operation is complete.
send() Method	flash.net.sendToURL()	
sendAndLoad() Method	flash.net.sendToURL()	The sendToURL() method sends a URL request to the server, but ignores the response. To receive the response, use

		flash.net.sendToURL().
toString() Method	Removed	
LocalConnection class	flash.net.LocalConnection	This class has been moved to the flash.net package.
LocalConnection Constructor	flash.net.LocalConnection.LocalConnection()	
allowDomain() EventHandler	flash.net.LocalConnection.allowDomain()	Changed to a regular method in ActionScript 3.0, no longer an event handler. Parameter changed to use the ...(rest) parameter format. Return value changed to void.
allowInsecureDomain() EventHandler	flash.net.LocalConnection.allowInsecureDomain()	Changed to a regular method in ActionScript 3.0, no longer an event handler. Parameter changed to use the ...(rest) parameter format. Return value changed to void.
close() Method	flash.net.LocalConnection.close()	
connect() Method	flash.net.LocalConnection.connect()	
domain() Method	flash.net.LocalConnection.domain	Changed to a property accessor.
onStatus() EventHandler	flash.net.LocalConnection dispatches event: status	In the new event model, callback functions are replaced by event objects.
send() Method	flash.net.LocalConnection.send()	Third parameter changed to use the ...(rest) parameter format. Return type changed to void.
Microphone class	flash.media.Microphone	This class has been moved to the flash.media package.
index Property[read-	flash.media.Microphone.index	Data type changed to

only]		uint.
rate Property[read-only]	flash.media.Microphone.rate	Data type changed to uint.
silenceTimeOut Property[read-only]	flash.media.Microphone.silenceTimeout	Change in capitalization to "Timeout." Data type changed to int.
get() Method	flash.media.Microphone.getMicrophone()	Name changed from get() to getMicrophone(). Data type of parameter changed to uint.
onActivity() EventHandler	flash.media.Microphone dispatches event: activity	In ActionScript 3.0, instead of invoking the onActivity event handler, this class dispatches an activity event.
onStatus() EventHandler	flash.media.Microphone dispatches event: status	In ActionScript 3.0, instead of invoking the onStatus event handler, this class dispatches a status event.
setGain() Method	flash.media.Microphone.gain	Combined gain property and setGain() method into a get/set property accessor named gain. Data type changed to uint.
setRate() Method	flash.media.Microphone.rate	Combined rate property and setRate() method into a get/set property accessor named rate. Data type changed to uint.
setSilenceLevel() Method	flash.media.Microphone.setSilenceLevel()	Data type of timeOut parameter changed to int. Capitalization of the timeOut parameter changed to timeout.
setUseEchoSuppression() Method	flash.media.Microphone.setUseEchoSuppression()	

Mouse class	flash.ui.Mouse	
addListener() Method	flash.events.EventDispatcher.addEventListener()	In the new ActionScript 3.0 event model, there is no need to have a class-specific addListener() method, because all display objects inherit the addEventListener() method from the EventDispatcher class.
hide() Method	flash.ui.Mouse.hide()	Changed to return void.
removeListener() Method	flash.events.EventDispatcher.removeEventListener()	In the new ActionScript 3.0 event model, there is no need to have a class-specific removeListener() method, because all display objects inherit the removeEventListener() method from the EventDispatcher class.
show() Method	flash.ui.Mouse.show()	Changed to return void.
onMouseDown Listener	flash.display.InteractiveObject dispatches event: mouseDown	Replaced in the new event model by a mouseDown event.
onMouseMove Listener	flash.display.InteractiveObject dispatches event: mouseMove	Replaced in the new event model by a mouseMove event.
onMouseUp Listener	flash.display.InteractiveObject dispatches event: mouseUp	Replaced in the new event model by a mouseUp event.
onMouseWheel Listener	flash.display.InteractiveObject dispatches event: mouseWheel	Replaced in the new event model by a mouseWheel event.
MovieClip class	flash.display.MovieClip	Many of the MovieClip methods have been moved to other classes in ActionScript 3.0. All event handlers have been replaced by

		event objects in the new event model.
_alpha Property	flash.display.DisplayObject.alpha	Moved to DisplayObject class and removed initial underscore from name.
blendMode Property	flash.display.DisplayObject.blendMode	
cacheAsBitmap Property	flash.display.DisplayObject.cacheAsBitmap	
_currentframe Property[read-only]	flash.display.MovieClip.currentFrame	Removed initial underscore from name.
_droptarget Property[read-only]	flash.display.Sprite.dropTarget	Moved to Sprite class, removed initial underscore from name, and changed to mixed case.
filters Property	flash.display.DisplayObject.filters	
focusEnabled Property	Removed	
_focusrect Property	flash.display.InteractiveObject.focusRect	Moved to InteractiveObject class, removed initial underscore from name, and changed to mixed case.
_framesloaded Property[read-only]	flash.display.MovieClip.framesLoaded	Removed initial underscore from name and changed to mixed case.
_height Property	flash.display.DisplayObject.height	Moved to DisplayObject class and removed initial underscore from name.
_highquality Property	Removed	
hitArea Property	flash.display.Sprite.hitArea	Moved to Sprite class.
_lockroot Property	Removed	
menu Property	Removed	
_name Property	flash.display.DisplayObject.name	Moved to DisplayObject class and removed initial underscore from name.
opaqueBackground	flash.display.DisplayObject.opaque	

Property	Background	
_parent Property	flash.display.DisplayObject.parent	Moved to DisplayObject class and removed initial underscore from name.
_quality Property	flash.display.Stage.quality	
_rotation Property	flash.display.DisplayObject.rotation	Moved to DisplayObject class and removed initial underscore from name.
scale9Grid Property	flash.display.DisplayObject.scale9Grid	
scrollRect Property	flash.display.DisplayObject.scrollRect	Changed to Rectangle data type.
_soundbuftime Property	flash.media.SoundMixer.bufferTime	Moved to SoundMixer class, which is used for global sound control, renamed without abbreviations, and removed initial underscore from name.
tabChildren Property	flash.display.DisplayObjectContainer.tabChildren	
tabEnabled Property	flash.display.InteractiveObject.tabEnabled	
tabIndex Property	flash.display.InteractiveObject.tabIndex	
_target Property[read-only]	Removed	
_totalframes Property[read-only]	flash.display.MovieClip.totalFrames	Changed to mixed case and removed initial underscore from name.
trackAsMenu Property	flash.display.MovieClip.trackAsMenu	
transform Property	flash.display.DisplayObject.transform	
_url Property[read-only]	flash.display.Loader.contentLoaderInfo	
useHandCursor Property	flash.display.Sprite.useHandCursor	
_visible Property	flash.display.DisplayObject.visible	Moved to DisplayObject class

		and removed initial underscore from name.
_width Property	flash.display.DisplayObject.width	Moved to DisplayObject class and removed initial underscore from name.
_x Property	flash.display.DisplayObject.x	Moved to DisplayObject class and removed initial underscore from name.
_xmouse Property[read-only]	flash.display.DisplayObject.mouseX	Moved to DisplayObject class, changed name to mouseX, and removed initial underscore from name.
_xscale Property	flash.display.DisplayObject.scaleX	Moved to DisplayObject class, changed name to scaleX, and removed initial underscore from name.
_y Property	flash.display.DisplayObject.y	Moved to DisplayObject class and removed initial underscore from name.
_ymouse Property[read-only]	flash.display.DisplayObject.mouseY	Moved to DisplayObject class, changed name to mouseY, and removed initial underscore from name.
_yscale Property	flash.display.DisplayObject.scaleY	Moved to DisplayObject class, changed name to scaleY, and removed initial underscore from name.
attachAudio() Method	Removed	If the audio source is a Microphone object, use NetStream.attachAudio() or Microphone.setLoopBack().

		If the audio source is an FLV file, use Video.attachNetStream() and a NetStream object.
attachBitmap() Method	Removed	In ActionScript 3.0, use addChild() to add child display objects.
attachMovie() Method	Removed	In ActionScript 3.0, use addChild() to add child display objects.
beginBitmapFill() Method	flash.display.Graphics.beginBitmapFill()	
beginFill() Method	flash.display.Graphics.beginFill()	Moved to Graphics class and changed data type of the first parameter to uint.
beginGradientFill() Method	flash.display.Graphics.beginGradientFill()	
clear() Method	flash.display.Graphics.clear()	
createEmptyMovieClip() Method	Removed	In ActionScript 3.0, use the new operator to create movie clips.
createTextField() Method	Removed	In ActionScript 3.0, use the new operator to create text fields.
curveTo() Method	flash.display.Graphics.curveTo()	
duplicateMovieClip() Method	Removed	In ActionScript 3.0, use the new operator to create a new instance.
endFill() Method	flash.display.Graphics.endFill()	
getBounds() Method	flash.display.DisplayObject.getBounds()	
getBytesLoaded() Method	flash.net.URLLoader.bytesLoaded	Moved to URLLoader class and data type changed from Number to int.
getBytesTotal() Method	flash.net.URLLoader.bytesTotal	Moved to URLLoader class and data type changed from Number to int.
getDepth() Method	flash.display.DisplayObjectContain	ActionScript 3.0

	er.getChildIndex()	provides direct access to the display list, so depth is handled differently.
getInstanceAtDepth() Method	flash.display.DisplayObjectContainer.getChildAt()	ActionScript 3.0 provides direct access to the display list, so depth is handled differently.
getNextHighestDepth() Method	flash.display.DisplayObjectContainer.addChild()	Not a direct equivalent, but the addChild() method adds a child behind all other children of the DisplayObjectContainer instance, so there is no need for a method that determines the next available depth.
getRect() Method	flash.display.DisplayObject.getRect()	
getSWFVersion() Method	flash.display.LoaderInfo.swfVersion	Moved to LoaderInfo class and changed data type to uint.
getTextSnapshot() Method	flash.display.DisplayObjectContainer.textSnapshot	
getURL() Method	flash.net.navigateToURL()	Replaced by the flash.net.navigateToURL() and flash.net.sentToURL() methods. Also see URLLoader class.
globalToLocal() Method	flash.display.DisplayObject.globalToLocal()	
gotoAndStop() Method	flash.display.MovieClip.gotoAndStop()	
hitTest() Method	flash.display.DisplayObject.hitTestObject()	
lineGradientStyle() Method	flash.display.Graphics.lineGradientStyle()	
lineStyle() Method	flash.display.Graphics.lineStyle()	
lineTo() Method	flash.display.Graphics.lineTo()	
loadMovie() Method	flash.display.Loader.load()	See Loader class.

loadVariables() Method	flash.net.URLLoader	Removed. See URLLoader class.
localToGlobal() Method	flash.display.DisplayObject.localToGlobal()	
moveTo() Method	flash.display.Graphics.moveTo()	
nextFrame() Method	flash.display.MovieClip.nextFrame()	
onData() EventHandler	flash.display.LoaderInfo dispatches event: complete	Replaced in the new event model by a complete event, which is dispatched when the download operation is complete but before any data is parsed.
onDragOut() EventHandler	flash.display.InteractiveObject dispatches event: mouseOut	Replaced in the new event model by a mouseOut event, after a call to InteractiveObject.setCapture().
onDragOver() EventHandler	flash.display.InteractiveObject dispatches event: mouseOver	Replaced in the new event model by a mouseOver event, after a call to InteractiveObject.setCapture().
onEnterFrame() EventHandler	flash.display.DisplayObject dispatches event: enterFrame	Replaced in the new event model by an enterFrame event.
onKeyDown() EventHandler	flash.display.InteractiveObject dispatches event: keyDown	Replaced in the new event model by a keyDown event.
onKeyUp() EventHandler	flash.display.InteractiveObject dispatches event: keyUp	Replaced in the new event model by a keyUp event.
onKillFocus() EventHandler	flash.display.InteractiveObject dispatches event: focusOut	Replaced in the new event model by a focusOut event.
onLoad() EventHandler	flash.display.LoaderInfo dispatches event: complete	Also see URLLoader class. A complete event is dispatched when the download operation is complete.
onMouseDown()	flash.display.InteractiveObject	Replaced in the new

EventHandler	dispatches event: mouseDown	event model by a mouseDown event.
onMouseMove() EventHandler	flash.display.InteractiveObject dispatches event: mouseMove	Replaced in the new event model by a mouseMove event.
onMouseUp() EventHandler	flash.display.InteractiveObject dispatches event: mouseUp	Replaced in the new event model by a mouseUp event.
onPress() EventHandler	flash.display.InteractiveObject dispatches event: mouseDown	Replaced in the new event model by a mouseDown event.
onRelease() EventHandler	flash.display.InteractiveObject dispatches event: mouseUp	Replaced in the new event model by a mouseUp event.
onReleaseOutside() EventHandler	flash.display.InteractiveObject dispatches event: mouseUp	Replaced in the new event model by a mouseUp event after a call to flash.display.InteractiveObject.setCapture().
onRollOut() EventHandler	flash.display.InteractiveObject dispatches event: mouseOut	Replaced in the new event model by a mouseOut event.
onRollOver() EventHandler	flash.display.InteractiveObject dispatches event: mouseOver	Replaced in the new event model by a mouseOver event.
onSetFocus() EventHandler	flash.display.InteractiveObject dispatches event: focusIn	Replaced in the new event model by a focusIn event.
onUnload() EventHandler	flash.display.LoaderInfo dispatches event: unload	Replaced in the new event model by an unload event.
play() Method	flash.display.MovieClip.play()	
prevFrame() Method	flash.display.MovieClip.prevFrame()	
removeMovieClip() Method	flash.display.DisplayObjectContainer.removeChild()	Removed. Call the removeChild() method of the parent display object container that contains the movie clip.
setMask() Method	flash.display.DisplayObject.mask	
startDrag() Method	flash.display.Sprite.startDrag()	

stop() Method	flash.display.MovieClip.stop()	
stopDrag() Method	flash.display.Sprite.stopDrag()	
swapDepths() Method	Removed	In ActionScript 3.0, you can achieve similar functionality by using the methods of the DisplayObjectContainer class, such as the addChildAt(), setChildIndex(), swapChildren(), and swapChildrenAt() methods.
unloadMovie() Method	flash.display.Loader.unload()	
MovieClipLoader class	flash.display.Loader	Replaced by the flash.display.Loader class.
MovieClipLoader Constructor	flash.display.Loader.Loader()	
addListener() Method	flash.events.EventDispatcher.addEventListener()	In the new event model, there is no need to have a class-specific addListener() method, because the class inherits the addEventListener() method from the EventDispatcher class.
getProgress() Method	flash.display.LoaderInfo dispatches event: progress	Replaced in the new event model by a progress event. Event objects of progress type contain properties named bytesLoaded and bytesTotal.
loadClip() Method	flash.display.Loader.load()	Replaced by the load() method of flash.display.Loader class.
removeListener() Method	flash.events.EventDispatcher.removeEventListener()	In the new event model, there is no need to have a class-specific

		removeListener() method, because the class inherits the removeEventListener() method from the EventDispatcher class.
unloadClip() Method	flash.display.Loader.unload()	Replaced by unload() method of flash.display.Loader class.
onLoadComplete Listener	flash.display.LoaderInfo dispatches event: complete	Replaced in the new event model by a complete event.
onLoadError Listener	flash.display.LoaderInfo dispatches event: ioError	Replaced in the new event model by an ioError event.
onLoadInit Listener	flash.display.LoaderInfo dispatches event: init	Replaced in the new event model by an init event.
onLoadProgress Listener	flash.display.LoaderInfo dispatches event: progress	Replaced in the new event model by a progress event.
onLoadStart Listener	flash.display.LoaderInfo dispatches event: open	Replaced in the new event model by an open event.
NetConnection class	flash.net.NetConnection	This class has been moved to the flash.net package.
NetConnection Constructor	flash.net.NetConnection.NetConnection()	
connect() Constructor	flash.net.NetConnection.connect()	ActionScript 3.0 version adds a ...(rest) parameter.
NetStream class	flash.net.NetStream	This class has been moved to the flash.net package.
bytesLoaded Property[read-only]	flash.net.NetStream.bytesLoaded	Data type changed to uint.
bytesTotal Property[read-only]	flash.net.NetStream.bytesTotal	Data type changed to uint.
currentFps	flash.net.NetStream.currentFPS	In ActionScript 3.0,

Property[read-only]		FPS is all uppercase.
onStatus() EventHandler	flash.net.NetStream dispatches event: netStatus	Replaced in the new event model by a netStatus event.
pause() Method	flash.net.NetStream.pause()	In ActionScript 3.0, the pause method does not take a parameter. Two new methods are available to achieve the same functionality: resume() and togglePause().
play() Method	flash.net.NetStream.play()	The name, start, len, and reset parameters are no longer valid; ...arguments is used instead.
setBufferTime() Method	flash.net.NetStream.bufferTime	In ActionScript 3.0, changed to read-write accessor property.
Number class	Number	
Number Constructor	Number.Number()	In ActionScript 3.0, the Number() constructor and the Number() global function have the same effect. Also, there is no difference between a Number object and a literal Number value.
Object class	Object	
__proto__ Property	Removed	In ActionScript 3.0, direct manipulation of the prototype chain is not allowed. To create a subclass, use the extends statement in the subclass declaration. For information about an object's inheritance tree and data type, use the new reflection API

		flash.utils.describeType().
__resolve Property	flash.utils.Proxy	Use the new Proxy class for similar functionality.
addProperty() Method	Removed	In ActionScript 3.0, accessor properties can be created directly using the keywords get and set.
registerClass() Method	Removed	In ActionScript 3.0, all classes are registered by default. If you are encoding an object using AMF, the class of the object is not preserved during the encoding process unless you use the flash.utils.registerClassAlias() function.
unwatch() Method	Removed	
watch() Method	Removed	Use accessor properties (get/set functions) or the flash.utils.Proxy class for similar functionality.
PrintJob class	flash.printing.PrintJob	
orientation Property[read-only]	flash.printing.PrintJob.orientation	This property now has a value from the PrintJobOrientation class.
pageHeight Property[read-only]	flash.printing.PrintJob.pageHeight	Data type changed to int.
pageWidth Property[read-only]	flash.printing.PrintJob.pageWidth	Data type changed to int.
paperHeight Property[read-only]	flash.printing.PrintJob.paperHeight	Data type changed to int.
paperWidth Property[read-only]	flash.printing.PrintJob.paperWidth	Data type changed to int.
PrintJob Constructor	flash.printing.PrintJob.PrintJob()	
addPage() Method	flash.printing.PrintJob.addPage()	In ActionScript 3.0,

		changed data types of parameters: First parameter target is a Sprite data type; second parameter printArea is a Rectangle data type; third parameter options is the new PrintJobOptions data type; and fourth parameter frameNum is an int data type.
send() Method	flash.printing.PrintJob.send()	
start() Method	flash.printing.PrintJob.start()	
Rectangle class		
containsRectangle() Method	flash.geom.Rectangle.containsRect()	Renamed for consistency.
security class	flash.system.Security	This class has been moved to the flash.system package.
Selection class	Removed	Methods of this class have been moved to other classes.
addListener() Method	flash.events.EventDispatcher.addEventListener()	In the new event model, there is no need to have a class-specific addListener() method, because any display object inherits the addEventListener() method from the EventDispatcher class.
getBeginIndex() Method	flash.text.TextField.selectionBeginIndex	Changed from method to accessor property and name changed to selectionBeginIndex.
getCaretIndex() Method	flash.text.TextField.caretIndex	Changed from method to accessor property and name changed to caretIndex.

getEndIndex() Method	flash.text.TextField.selectionEndIndex	Changed from method to accessor property and name changed to selectionEndIndex.
getFocus() Method	flash.display.Stage.focus	Changed from method to property accessor and name changed to focus. In ActionScript 2.0 the data type of the return value is String, but in ActionScript 3.0 the property has the InteractiveObject data type.
removeListener() Method	flash.events.EventDispatcher.removeEventListener()	In the new event model, there is no need to have a class-specific removeListener() method, because display objects inherit the removeEventListener() method from the EventDispatcher class.
setFocus() Method	flash.display.Stage.focus	Changed from method to accessor property and name changed to focus. In ActionScript 2.0 the data type of the return value is String, but in ActionScript 3.0 the property has the InteractiveObject data type.
setSelection() Method	flash.text.TextField.setSelection()	Both parameters changed from Number to uint data type.
onSetFocus Listener	flash.display.InteractiveObject dispatches event: focusIn	Replaced in the new event model by a focusIn event.
SharedObject class	flash.net.SharedObject	This class has been moved to the flash.net package.

flush() Method	flash.net.SharedObject.flush()	This method no longer returns a Boolean value. If the flush fails, Flash Player throws an exception; if the flush succeeds or is pending user interaction, Flash Player returns a string "flushed" or "pending". Also, the data type of the minDiskSpace parameter changed to int.
getSize() Method	flash.net.SharedObject.size	Changed to accessor property. Data type changed to uint.
onStatus() EventHandler	flash.net.SharedObject dispatches event: netStatus	Replaced in the new event model by a netStatus event.
Sound class	flash.media.Sound	This class has been moved to the flash.media package.
checkPolicyFile	flash.media.SoundChannel.stop()	Replaced by the flash.media.SoundChannel.stop() method.
duration Property[read-only]	flash.media.Sound.length	
id3 Property[read-only]	flash.media.Sound.id3	Data type changed from Object to ID3Info. ID3Info is a new class that contains the ID3 properties. Also, the spelling of the songname property changed to songName.
position Property[read-only]	flash.media.SoundChannel.position	Moved to the SoundChannel class.
attachSound() Method	Removed	Create an instance of a Sound subclass that is associated with sound data; for example, by using new Sound() instead.
getBytesLoaded()	flash.media.Sound.bytesLoaded	Changed to accessor

Method		property and data type changed to uint.
getBytesTotal() Method	flash.media.Sound.bytesTotal	Changed to property accessor and data type changed to uint.
getPan() Method	flash.media.SoundTransform.pan	Changed to accessor property and moved to the SoundTransform class.
getTransform() Method	flash.media.SoundMixer.soundTransform	Changed to accessor property and data type changed to SoundTransform.
getVolume() Method	flash.media.SoundTransform.volume	Set the flash.media.SoundTransform.volume property to control sound volume.
loadSound() Method	flash.media.Sound.load()	The first parameter changed from a simple URL string to a URLRequest object. The second parameter changed from a Boolean value representing whether sound begins playing as soon as possible to a SoundLoaderContext object.
onID3() EventHandler	flash.media.Sound dispatches event: id3	Replaced in the new event model by an id3 event.
onLoad() EventHandler	flash.media.Sound dispatches event: complete	Replaced in the new event model by a complete event.
onSoundComplete() EventHandler	flash.media.SoundChannel dispatches event: soundComplete	Replaced in the new event model by a soundComplete event.
setPan() Method	flash.media.SoundTransform.pan	Changed to accessor property and moved to SoundTransform class.
setTransform() Method	flash.media.SoundMixer.soundTransform	Changed to accessor property and data type changed to

		SoundTransform.
setVolume() Method	flash.media.SoundChannel	Removed. Use flash.media.SoundChannel.leftPeak and flash.media.SoundChannel.rightPeak to monitor the amplitude of a sound channel.
start() Method	flash.media.Sound.play()	The loops parameter data type changed from Number to int. Added a third parameter, sndTransform, to specify the initial sound transform to be used by the sound channel.
stop() Method	flash.media.SoundChannel.stop()	
Stage class	flash.display.Stage	This class has been moved to the flash.display package. In ActionScript 3.0, the Stage is no longer a global object. You access the Stage by using the new DisplayObject.stage property.
align Property	flash.display.Stage.align	
height Property	flash.display.Stage.stageHeight	Name changed from height to stageHeight so that it does not conflict with the flash.display.DisplayObject.height property.
scaleMode Property	flash.display.Stage.scaleMode	
showMenu Property	flash.display.Stage.showDefaultContextMenu	Name changed to better reflect which menu is shown.
width Property	flash.display.Stage.stageWidth	Name changed from width to stageWidth so that it does not conflict with the flash.display.DisplayO

		bject.width property.
addListener() Method	flash.events.EventDispatcher.addEventListener()	In the new event model, there is no need to have a class-specific addListener() method, because the class inherits the addEventListener() method from the EventDispatcher class.
removeListener() Method	flash.events.EventDispatcher.removeEventListener()	In the new event model, there is no need to have a class-specific removeListener() method, because the class inherits the removeEventListener() method from the EventDispatcher class.
onResize Listener	flash.display.Stage dispatches event: resize	Replaced in the new event model by a resize event.
String class	String	Adds support for regular expressions with three new methods: match(), replace(), and search().
concat() Method	String.concat()	Parameter changed to ...(rest) parameter format.
StyleSheet class	flash.text.StyleSheet	This class has been moved to the flash.text package. The load() and onLoad() members have been removed, and some private functions and variables have been added.
StyleSheet Constructor	flash.text.StyleSheet.StyleSheet()	

clear() Method	flash.text.StyleSheet.clear()	
getStyle() Method	flash.text.StyleSheet.getStyle()	Parameter name changed to n.
getStyleNames() Method	flash.text.StyleSheet.styleNames	Changed to accessor property.
load() Method	flash.net.URLLoader.load()	Use the new URLLoader and URLRequest classes for loading URLs.
onLoad() EventHandler	flash.net.URLLoader dispatches event: complete	Replaced in the new event model by a complete event.
parseCSS() Method	flash.text.StyleSheet.parseCSS()	In ActionScript 3.0, returns void instead of a Boolean value.
setStyle() Method	flash.text.StyleSheet.setStyle()	Parameter name changed to n and style to s.
transform() Method	flash.text.StyleSheet.transform()	
System class	flash.system.System	
exactSettings Property	flash.system.Security.exactSettings	Moved to the flash.System.Security class.
useCodepage Property	flash.system.System.useCodePage	In ActionScript 3.0, the letter 'P' in useCodePage is uppercase.
onStatus() EventHandler	Removed	
setClipboard() Method	flash.system.System.setClipboard()	
showSettings() Method	flash.system.Security.showSettings()	
TextField class	flash.text.TextField	This class has been moved to the flash.text package.
_alpha Property	flash.display.DisplayObject.alpha	This property is now inherited from the DisplayObject class. Removed the initial underscore.

antiAliasType Property	flash.text.TextField.antiAliasType	
autoSize Property	flash.text.TextField.autoSize	
background Property	flash.text.TextField.background	
backgroundColor Property	flash.text.TextField.backgroundColor	
border Property	flash.text.TextField.border	
borderColor Property	flash.text.TextField.borderColor	In ActionScript 3.0, returns a uint instead of a Number.
bottomScroll Property[read-only]	flash.text.TextField.bottomScrollV	In ActionScript 3.0, returns a uint instead of a Number.
condenseWhite Property	flash.text.TextField.condenseWhite	
embedFonts Property	flash.text.TextField.embedFonts	
filters Property	flash.display.DisplayObject.filters	
gridFitType Property	flash.text.TextField.gridFitType	
_height Property	flash.display.DisplayObject.height	This property is now inherited from the DisplayObject class. Removed the initial underscore.
_highquality Property	flash.display.Stage.quality	Removed. Replaced by the quality property of the Stage class.
hscroll Property	flash.text.TextField.scrollH	Data type changed from Number to uint. Name changed from hscroll to scrollH.
html Property	flash.text.TextField.htmlText	Removed. In ActionScript 3.0, all text fields are treated as HTML text fields. Use the TextField.htmlText property to set HTML text.
htmlText Property	flash.text.TextField.htmlText	
length Property[read-only]	flash.text.TextField.length	Data type changed from Number to uint.
maxChars Property	flash.text.TextField.maxChars	Data type changed from Number to uint.

maxhscroll Property[read-only]	flash.text.TextField.maxScrollH	Data type changed from Number to uint.
maxscroll Property[read-only]	flash.text.TextField.maxScrollV	Data type changed from Number to uint. Name changed to use uppercase S and to add the letter V to represent vertical scrolling.
menu Property	flash.display.InteractiveObject.contextMenu	This property is now inherited from the InteractiveObject class.
mouseWheelEnabled Property	flash.text.TextField.mouseWheelEnabled	
multiline Property	flash.text.TextField.multiline	
_name Property	flash.display.DisplayObject.name	This property is now inherited from the DisplayObject class. Removed the initial underscore.
_parent Property	flash.display.DisplayObject.parent	This property is now inherited from the DisplayObject class. Removed the initial underscore. Data type changed from MovieClip to DisplayObjectContainer.
password Property	flash.text.TextField.displayAsPassword	Renamed property for consistency.
_quality Property	flash.display.Stage.quality	Moved to Stage class.
restrict Property	flash.text.TextField.restrict	
_rotation Property	flash.display.DisplayObject.rotation	This property is now inherited from the DisplayObject class. Removed the initial underscore.
scroll Property	flash.text.TextField.scrollV	Data type changed from Number to uint and name changed from scroll to scrollV.
selectable Property	flash.text.TextField.selectable	
sharpness Property	flash.text.TextField.sharpness	

_soundbuftime Property	flash.media.SoundMixer.bufferTime	Properties and methods for global sound control in a SWF file are now in the flash.media.SoundMixer class.
styleSheet Property	flash.text.TextField.styleSheet	
tabEnabled Property	flash.display.InteractiveObject.tabEnabled	This property is now inherited from the InteractiveObject class.
tabIndex Property	flash.display.InteractiveObject.tabIndex	This property is now inherited from the InteractiveObject class.
_target Property[read-only]	Removed	
text Property	flash.text.TextField.text	
textColor Property	flash.text.TextField.textColor	Data type changed from Number to uint.
textHeight Property	flash.text.TextField.textHeight	
textWidth Property	flash.text.TextField.textWidth	
thickness Property	flash.text.TextField.thickness	
type Property	flash.text.TextField.type	
_url Property[read-only]	flash.display.LoaderInfo.url	
variable Property	Removed	
_visible Property	flash.display.DisplayObject.visible	This property is now inherited from the DisplayObject class. Removed the initial underscore.
_width Property	flash.display.DisplayObject.width	This property is now inherited from the DisplayObject class. Removed the initial underscore.
wordWrap Property	flash.text.TextField.wordWrap	
_x Property	flash.display.DisplayObject.x	This property is now inherited from the DisplayObject class. Removed the initial underscore.
_xmouse Property[read-only]	flash.display.DisplayObject.mouseX	This property is now inherited from the

		DisplayObject class. Removed the initial underscore.
_xscale Property	flash.display.DisplayObject.scaleX	This property is now inherited from the DisplayObject class. Removed the initial underscore.
_y Property	flash.display.DisplayObject.y	This property is now inherited from the DisplayObject class. Removed the initial underscore.
_ymouse Property[read-only]	flash.display.DisplayObject.mouseY	This property is now inherited from the DisplayObject class. Removed the initial underscore.
_yscale Property	flash.display.DisplayObject.scaleY	This property is now inherited from the DisplayObject class. Removed the initial underscore.
addListener() Method	flash.events.EventDispatcher.addEventListener()	In the new event model, there is no need to have a class-specific addListener() method, because the class inherits the addEventListener() method from the EventDispatcher class.
getDepth() Method	flash.display.DisplayObjectContainer	Removed. Use the methods of the DisplayObjectContainer class to ascertain text field depth.
getFontList() Method	flash.text.Font.enumerateFonts()	Removed. Use Font.enumerateFonts() with the enumerateDeviceFonts parameter set to true.
getNewTextFormat() Method	flash.text.TextField.defaultTextFormat	Name changed from getNewTextFormat to defaultTextFormat.

		Changed from method to accessor property.
getTextFormat() Method	flash.text.TextField.getTextFormat()	Data type of both parameters changed from Number to uint.
onChanged() EventHandler	flash.text.TextField dispatches event: change	Replaced in the new event model by a change event.
onKillFocus() EventHandler	flash.display.InteractiveObject dispatches event: focusOut	Replaced in the new event model by a focusOut event.
onScroller() EventHandler	flash.text.TextField dispatches event: scroll	Replaced in the new event model by a scroll event.
onSetFocus() EventHandler	flash.display.InteractiveObject dispatches event: focusIn	Replaced in the new event model by a focusIn event.
removeListener() Method	flash.events.EventDispatcher.removeEventListener()	In the new event model, there is no need to have a class-specific removeListener() method, because the class inherits the removeEventListener() method from the EventDispatcher class.
removeTextField() Method	flash.display.DisplayObjectContainer.removeChild()	Removed. Call the removeChild() method of the parent display object container that contains the text field.
replaceSel() Method	flash.text.TextField.replaceSelectedText()	Name changed from replacesel() to replaceSelectedText(). Replaced the newText parameter with a string value.
replaceText() Method	flash.text.TextField.replaceText()	Data types of first two parameters changed from Number to uint.
setNewTextFormat() Method	flash.text.TextField.defaultTextFormat	Name changed from setNewTextFormat to defaultTextFormat. Changed from method

		to accessor property.
setTextFormat() Method	flash.text.TextField.setTextFormat()	Order of parameters changed. Index parameters data type changed from Number to int.
TextFormat class	flash.text.TextFormat	This class has been moved to the flash.text package.
align Property	flash.text.TextFormat.align	
blockIndent Property	flash.text.TextFormat.blockIndent	In ActionScript 3.0, data type changed to Object because one of the possible values is null, which is not a member of the Number data type in ActionScript 3.0.
bold Property	flash.text.TextFormat.bold	In ActionScript 3.0, data type changed to Object because one of the possible values is null, which is not a member of the Boolean data type in ActionScript 3.0.
bullet Property	flash.text.TextFormat.bullet	In ActionScript 3.0, data type changed to Object because one of the possible values is null, which is not a member of the Boolean data type in ActionScript 3.0.
color Property	flash.text.TextFormat.color	In ActionScript 3.0, data type changed to Object because one of the possible values is null, which is not a member of the Number data type in ActionScript 3.0.
font Property	flash.text.TextFormat.font	
indent Property	flash.text.TextFormat.indent	In ActionScript 3.0,

		data type changed to Object because one of the possible values is null, which is not a member of the Number data type in ActionScript 3.0.
italic Property	flash.text.TextFormat.bullet	In ActionScript 3.0, data type changed to Object because one of the possible values is null, which is not a member of the Boolean data type in ActionScript 3.0.
kerning Property	flash.text.TextFormat.kerning	In ActionScript 3.0, data type changed to Object because one of the possible values is null, which is not a member of the Boolean data type in ActionScript 3.0.
leading Property	flash.text.TextFormat.leading	In ActionScript 3.0, data type changed to Object because one of the possible values is null, which is not a member of the Number data type in ActionScript 3.0.
leftMargin Property	flash.text.TextFormat.leftMargin	In ActionScript 3.0, data type changed to Object because one of the possible values is null, which is not a member of the Number data type in ActionScript 3.0.
letterSpacing Property	flash.text.TextFormat.letterSpacing	In ActionScript 3.0, data type changed to Object because one of the possible values is null, which is not a member of the Number data type in

		ActionScript 3.0.
rightMargin Property	flash.text.TextFormat.rightMargin	In ActionScript 3.0, data type changed to Object because one of the possible values is null, which is not a member of the Number data type in ActionScript 3.0.
size Property	flash.text.TextFormat.size	In ActionScript 3.0, data type changed to Object because one of the possible values is null, which is not a member of the Number data type in ActionScript 3.0.
underline Property	flash.text.TextFormat.underline	In ActionScript 3.0, data type changed to Object because one of the possible values is null, which is not a member of the Boolean data type in ActionScript 3.0.
url Property	flash.text.TextFormat.url	
TextFormat Constructor	flash.text.TextFormat.TextFormat()	The size, color, bold, italic, underline, url, leftMargin, rightMargin, indent, and leading parameters have all been converted to objects.
getTextExtent() Method	Removed	Use the properties of flash.text.TextField for the measurements of a field containing a line of text, and use flash.text.TextLineMetrics for the measurements of the content within the text field.
TextRenderer class	flash.text.TextRenderer	Location changed.

		Moved to flash.text package.
maxLevel Property	flash.text.TextRenderer.maxLevel	Defined as a uint in ActionScript 3.0.
setAdvancedAntialiasingTable() Method	flash.text.TextRenderer.setAdvancedAntiAliasingTable()	The fontStyle and colorType parameter values can now be set using the FontStyle and TextColorType constants, respectively. The advancedAntiAliasingTable parameter now takes an array of one or more CSMSettings objects.
TextSnapshot class	flash.text.TextSnapshot	This class has been moved to the flash.text package. Several parameters have changed, as well as some method names and some return types.
findText() Method	flash.text.TextSnapshot.findText()	Name of the startIndex parameter changed to beginIndex. Data type of the startIndex parameter changed from Number to int.
getCount() Method	flash.text.TextSnapshot.charCount	Changed from method to accessor property. Data return type changed from Number to uint.
getSelected() Method	flash.text.TextSnapshot.getSelected()	Data type of parameters changed from Number to uint and names changed from start and end to beginIndex and EndIndex.
getSelectedText() Method	flash.text.TextSnapshot.getSelectedText()	In ActionScript 3.0, the parameter has a default value of false.

getText() Method	flash.text.TextSnapshot.getText()	Data type of start and end parameters changed from Number to uint and names changed from start and end to beginIndex and endIndex.
getTextRunInfo() Method	flash.text.TextSnapshot.getTextRunInfo()	Data type of parameters changed from Number to uint.
hitTestTextNearPos() Method	flash.text.TextSnapshot.hitTestTextNearPos()	Name of the closeDist parameter changed to maxDistance; now has a default value = 0.
setSelectColor() Method	flash.text.TextSnapshot.setSelectColor()	Data type of parameter changed from Number to uint and has a default value = 0xFFFF00.
setSelected() Method	flash.text.TextSnapshot.setSelected()	Data type of start and end parameters changed from Number to uint and names changed from start and end to beginIndex and endIndex.
Video class	flash.media.Video	This class has been moved to the flash.media package. Video objects can now be created dynamically in ActionScript with the Video() constructor. Attach a video stream to the Video object by using attachCamera() or attachNetStream().
_alpha Property	flash.display.DisplayObject.alpha	This property is inherited from the DisplayObject class. Removed the initial underscore.
deblocking Property	flash.media.Video.deblocking	Data type changed from Number to int.

_height Property	flash.display.DisplayObject.height	This property is inherited from the DisplayObject class Removed the initial underscore.
height Property[read-only]	flash.media.Video.videoHeight	Data type changed from Number to int.
_name Property	flash.display.DisplayObject.name	This property is inherited from the DisplayObject class. Removed the initial underscore.
_parent Property	flash.display.DisplayObject.parent	This property is inherited from the DisplayObject class. Removed the initial underscore.
_rotation Property	flash.display.DisplayObject.rotation	This property is inherited from the DisplayObject class. Removed the initial underscore.
smoothing Property	flash.media.Video.smoothing	
_visible Property	flash.display.DisplayObject.visible	This property is inherited from the DisplayObject class. Removed the initial underscore.
_width Property	flash.display.DisplayObject.width	This property is inherited from the DisplayObject class. Removed the initial underscore.
width Property[read-only]	flash.media.Video.videoWidth	Data type changed from Number to int.
_x Property	flash.display.DisplayObject.x	This property is inherited from the DisplayObject class. Removed the initial underscore.
_xmouse Property[read-only]	flash.display.DisplayObject.mouseX	This property is inherited from the DisplayObject class. Removed the initial underscore.

_xscale Property	flash.display.DisplayObject.scaleX	This property is inherited from the DisplayObject class. Removed the initial underscore.
_y Property	flash.display.DisplayObject.y	This property is inherited from the DisplayObject class. Removed the initial underscore.
_ymouse Property[read-only]	flash.display.DisplayObject.mouseY	This property is inherited from the DisplayObject class. Removed the initial underscore.
_yscale Property	flash.display.DisplayObject.scaleY	This property is inherited from the DisplayObject class. Removed the initial underscore.
attachVideo() Method	flash.media.Video.attachNetStream()	To specify a video stream from a camera object, use flash.media.Video.attachCamera().
clear() Method	flash.media.Video.clear()	
XML class	flash.xml.XMLDocument	This class has been moved to the flash.xml package and its name has been changed to XMLDocument to avoid conflict with the new top-level XML class that implements ECMAScript for XML (E4X).
contentType Property	flash.net.URLRequest.contentType	
docTypeDecl Property	flash.xml.XMLDocument.docTypeDecl	
idMap Property	flash.xml.XMLDocument.idMap	
ignoreWhite Property	flash.xml.XMLDocument.ignoreWhite	
loaded Property	Removed	File loading

		functionality was removed from the XMLDocument class. Use URLLoader instead.
status Property	Removed	Parse failures are now reported by exceptions.
xmlDecl Property	flash.xml.XMLDocument.xmlDecl	
XML Constructor	flash.xml.XMLDocument.XMLDocument()	
addRequestHeader() Method	flash.net.URLRequest.requestHeaders	
createElement() Method	flash.xml.XMLDocument.createElement()	
createTextNode() Method	flash.xml.XMLDocument.createTextNode()	
getBytesLoaded() Method	flash.net.URLLoader.bytesLoaded	File loading functionality was removed from the XMLDocument class. Use URLLoader instead.
getBytesTotal() Method	flash.net.URLLoader.bytesTotal	File loading functionality was removed from the XMLDocument class. Use URLLoader instead.
load() Method	Removed	File loading functionality was removed from the XMLDocument class (formerly the XML class in ActionScript 2.0). Use URLLoader instead.
onData() EventHandler	flash.net.URLLoader dispatches event: complete	File loading functionality was removed from the XMLDocument class. Use URLLoader instead. Replaced in the new event model by a complete event.

onHTTPStatus() EventHandler	flash.net.URLLoader dispatches event: httpStatus	File loading functionality was removed from the XMLDocument class. Use URLLoader instead. Replaced in the new event model by an httpStatus event.
onLoad() EventHandler	flash.net.URLLoader dispatches event: complete	File loading functionality was removed from the XMLDocument class. Use URLLoader instead. Replaced in the new event model by a complete event.
parseXML() Method	flash.xml.XMLDocument.parseXML()	
send() Method	Removed	Send functionality was removed from the XMLDocument class (formerly the XML class in ActionScript 2.0). Use the functions and classes of the flash.net package instead.
sendAndLoad() Method	Removed	Send and load functionality were removed from the XMLDocument class (formerly the XML class in ActionScript 2.0). Use URLRequest and URLLoader instead.
XMLNode class	flash.xml.XMLNode	Location changed. This class has been moved to the flash.xml package.
nodeType Property[read-only]	flash.xml.XMLNode.nodeType	Data type changed from Number to uint.
XMLNode Constructor	flash.xml.XMLNode.XMLNode()	Data type of the type parameter changed

		from Number to uint.
XMLSocket class	flash.net.XMLSocket	This class has been moved to the flash.net package.
XMLSocket Constructor	flash.net.XMLSocket.XMLSocket()	Added two optional parameters to specify host and port.
connect() Method	flash.net.XMLSocket.connect()	Data type of the port parameter changed to int.
onClose() EventHandler	flash.net.XMLSocket dispatches event: close	Replaced in the new event model by a close event.
onConnect() EventHandler	flash.net.XMLSocket dispatches event: connect	Replaced in the new event model by a connect event.
onData() EventHandler	flash.net.XMLSocket dispatches event: data	Replaced in the new event model by a data event.
onXML() EventHandler	Removed	In ActionScript 3.0, only the data event is dispatched, so that you can choose whether to use E4X or the legacy XML (XMLDocument class) parser. The old onXML event handler was called after XML was parsed, so it doesn't make sense in ActionScript 3.0 because you can now choose between the XML (E4X) class and the XMLDocument (legacy) class to parse the XML.

ملحق تعلم لغة ال#c للأستخدامها فى عمل تطبيقات الانترنت بأستخدام ASPX و ربطها ببرنامج الفلاش

مقدمة عامة

ما هي الإنترنت؟

تسمى أجهزة الكمبيوتر التي تستضيف مواقع الإنترنت Web Sites بالخوادم Servers، حيث تمنح الخدمات للعملاء Clients الذين يدخلون هذه المواقع.. وطبعا تكون هذه الأجهزة خاصة مثل أجهزة Main Frames، وهـي أجهـزة كمبيوتر أكبر في الحجم والسرعة ومساحة التخزين من الكمبيوتر الشخصي.

ولكن ما هو الإنترنت؟

إنّه عبارة عن شبكة كمبيوتر هائلة، تستخدم بروتوكولات معينة للتفـاهم فيمـا بـين أجزائهـا.. ومـن أهـم هـذه البروتوكولات، Transmission Control Protocol/Internet Protocol (TCP/IP)، وهو بروتوكـول بـسيط، قادر على التعامل مع كلّ أجهزة الكمبيوتر وكلّ أنظمة التشغيل.

ولا تجزع هكذا.. لسنا بحاجة لشرح هذا البرتوكول ولا سواه.. إن كلّ هذه التفاصيل تتم في الخفاء بعيدا عنك، وكـلّ ما عليك لاستغلال الإنترنت، هو وجود متصفّح الإنترنت Internet Explorer على جهازك، حيث يقوم بفتح مواقـع الإنترنت وتحميل الصفحات المطلوبة إلى جهازك (هذا بافتراض أن جهازك يحتوي على كـارت إنترنـت وممّـصل بالهاتف).

<u>لغة HTML:</u>

إن HTML هي اللغة التي يتخاطب بها الخوادم والعملاء لنقل الوثائق على الإنترنت.

وليست الوثيقة إلا ملفّا نصيا Text file، يحتوى على مراجع تشير إلى بعض الصور ليتم عرضها، وبعض الأصوات ليتم تشغيلها، وبعض الوصلات Links تشير إلى صفحات ووثائق أخرى.

ويمكنك أن تنشئ صفحات HTML بأي برنامج محرر نصوص مثل Notepad، حيث تكتب فيه أوامر اللغة يدويا.. لقد كان هذا هو الوضع قبل إصدار التطبيقات المسماة WYSIWYG (ما تراه هو ما تحـصل عليـه what you see is what you get)، مثل تطبيق

Microsoft FrontPage، Microsoft Office 2000.. في هذه التطبيقات تـستخدم واجهـةً مرئيـةً لإدراج عناصر الصفحة وتنسيق ألوانها وخطوطها وصورها، دون أن تكتب حرفا واحدا من لغة HTML.

وتوجد صفحات الإنترنت على الخوادم، التي تعرضها لكلّ من يطلبها، حيث يمكن في هذه الحالة أن يحفظ نسخة منها على جهازه.. ولكلّ خادم عنوان يسمى Uniform Resource Locator (URL)، يكون على شاكلة:

http://www.example.com

حيث http هو اسم البروتوكول المستخدم في التعامل مع الخادم، و www.example.com هو عنوان الخادم على الإنترنت.

إن لكلّ جهاز على الإنترنت عنوان برتوكول الإنترنت Internet Protocol (IP) Address، يكون على شـاكلة 193.22.103.18.. ونظرا لأنّه من الصعب علينا أن نحفظ مثل هذه الأرقام، فإن كلّ خادم يحتفظ بجدول به هـذه

الأرقام والعناوين النصية التي تناظرها.. وعندما تقوم بكتابة العنوان www.example.com فإن الخادم يبحث عن الرقم المناظر له ويستخدمه.

عناصر الكود في لغة HTML:

تتكون لغة HTML من مجموعة من عناصر تنسيق النصوص Text-formatting Tags، توضع كلّ منها بين علامتى < >.. وفي الغالب يظهر كلّ عنصر تنسيق Tag في صورة زوج: عنصر التنسيق الأول يمثّل بداية التنسيق، والآخر يمثّل نهايته.. فمثلا، لكي تعرض بعض الكلمات بخطّ سميك Bold، ضعها بين زوج من عنصر التنسيق <B> كالتالي:

Some <B>words</B> in the sentence are formatted in <B>bold</B>.

طبعا لاحظت أن عنصر تنسيق النهاية يسبقه الرمز "/"، (</B>).

ولترى مثالا لصفحة انترنت، انسخ الجمل التالية إلى ملفّ نص عادي، واحفظه.. غير امتداد الملفّ إلى HTM. .. انقر الملفّ بعد ذلك مرتين بالفأرة، ليتم عرضه في متصفّح الإنترنت:

```
<HTML>
<HEAD>
<TITLE>هذا هو عنوان الصفحة</TITLE>
</HEAD>
<BODY>
هذا هو نص الصفحة: <B>مرحبا بك</B>
</BODY>
</HTML>
```

وتستخدم بعض السمات Attributes مع عناصر التنسيق، وهي بعض الكلمات الخاصة، التي تؤدى وظيفة محددة مع عنصر التنسيق.. فعلى سبيل المثال، يستخدم عنصر التنسيق <A> لإدراج وصلة فائقة Hyperlink في الوثيقة.. هذا العنصر يستخدم السمة HREF.. انظر للجملة التالية، التي تضيف للصفحة وصلة لموقع ميكروسوفت على الإنترنت:

This <A HREF="http://www.microsoft.com">link</A> leads to Microsoft's home page.

إن النص الموجود بين علامتي <A> </A> سيعرض على أنّه وصلة للتنقل بين الصفحات.. وتخبر السمة HREF المتصفّح بالعنوان URL الذي سيتم الانتقال إليه عند ضغط الوصلة.

ولمزيد من التفاصيل حول لغة HTML، يمكنك الرجوع للملحق ٤، في نهاية هذا الكتاب.

أدوات HTML:

تتعامل لغة HTML مع بعض الأدوات البسيطة، مثل مربعات النص Text Boxes، أزرار التحويل Radio Buttons، مربعات الاختيار Check Boxes، الأزرار Buttons، وغيرها.

ويتم عرض هذه الأدوات على صفحة الإنترنت، حيث يمكن للمستخدم أن يختار أو يدخل بعض القيم ثم يضغط زر إرسالها إلى الخادم.. ويقوم الخادم بمعالجة هذه البيانات، وإعداد صفحة جديدة فيها ناتج المعالجة، حيث يرسلها ليتم عرضها على جهاز المستخدم.

ويستخدم ASP.NET (ASP هي اختصار لتعبير "صفحة الخادم النشيطة Active Server Page) نفس هذا السيناريو.

لغة XHTML:

في عام ٢٠٠٠ ظهر أول إصدار من لغة HTML الممددة Extensible، وذلك حتّى تتجاوب مع قواعد لغة XML.. وتستخدم هذه اللغة الجديدة معظم عناصر تنسيق HTML، ولكن مع بعض القيود:

- يجب كتابة أسماء عناصر التنسيق بحروف صغيرة Small
- يجب وضع قيم السمات بين علامتي تنصيص.
- يجب إغلاق كلّ مقطع تنسيق بعنصر نهاية التنسيق.

لماذا ASP.NET ؟

العيبُ الرئيسي في صفحات الإنترنت هي أنّها ثابتة المحتوى، مما يعنى صعوبة تحديثها.. هنا كان لا بد من ظهور تقنية Common Gateway Interface (CGI)، التي تسمح بإضافة محتويات مرنة إلى الإنترنت، بحيث تكون قابلة للتعديل.. فمثلا، يمكن للعميل أن يستعلم عن قائمة الأسعار الخاصة ببعض المنتجات اليوم.. في هذه الحالة لن يعيد الخادم صفحة HTML ثابتة المحتوى، لكنّه سينفّذ تطبيقا ما يجمع المعلومات المطلوبة، وبعد ذلك يحولها لتنسيق صفحة HTML ويرسلها للعميل.. وبهذا يرى العميل أحدث المعلومات.

ولكن تتبقّى مشكلة التعقيد في كتابة تطبيقات CGI وتطويرها.. هنا قدمت ميكروسوفت الحلّ في صورة صفحة الخادم النشطة ASP، وهي عبارة عن صفحة إنترنت تحتوي على برنامج مكتوب في الغالب بلغة VBScript، بحيث يمكنه أن يتفاعل مع العميل ومع المكونات الأخرى على الخادم.. فمثلا، يمكن لصفحة ASP أن تتفاعل مع قواعد البيانات للحصول على بعض البيانات المطلوبة.

والآن نحن نتعامل مع أحدث إصدار من ASP هو ASP.NET، الذي يمكّنك من كتابة برامج تعمل على الخادم بلغة C#.NET.

إن هناك ميزات هائلة لاستخدام C#.NET في كتابة هذه البرامج، فبخلاف توفير وقت التعلّم علينا، وسهولة البرمجة بـ C#، فإن ذلك يجعل التطبيق يعمل بسرعةٍ أكبر على الخادم.. هذا بالإضافة إلى استغلال كلّ إمكانيات بيئة التطوير المتكاملة IDE الخاصة بـ VS.NET، واستخدام مجموعة جديدة من أدوات الإنترنت Web Controls التي صممت لعلاج القصور في أدوات HTML.

أما الميزة الكبرى، فهي إحساسك بأنّك تكتب كود C# عادي تماما، مبتعدا معظم الوقت عن مفاهيم HTML و XML بل و ASP نفسه.

ملحوظة:

ليست لغة C# وحدها هي التي تستطيع بها كتابة تطبيقات ASP.NET، بل إن بإمكانك كتابتها ذلك بأي لغة تعمل على VS.NET مثل VB، J#، وCOBOL، لإلى آخر قائمة اللغات التي تنضم كلّ حينٍ إلى VS.NET.

إن تطبيقك يعمل على الخادم حيث توجد أدوات الإنترنت Web controls.. فكيف إذن تظهر هذه الأدوات على جهاز العميل؟

في هذه الحالة يقوم ASP.NET بتحويل أدوات الإنترنت إلى أدوات HTML وإرسالها كـصفحة HTML لجهـاز العميل ليتم عرضها.

الخلاصة، أنّك تتعامل مع نسخة رائعة من ASP.NET تشعرك بأنّك تكتب تطبيق ويندوز عادي، بينما تقوم هي فـي الخلفية بإنتاج صفحات الإنترنت المطلوبة.

التفاعل مع تطبيق الإنترنت:

هذا هو ما يحدث عند تنفيذ تطبيق إنترنت:

- يتركّب تطبيق الإنترنت من نموذج إنترنت أو أكثر.. ويتكون النموذج من كود HTML وكود C#.. وبينما يـتم تنفيذ كود C# على الخادم، فإن كود HTML يتم إرساله إلى متصفّح العميل ليقوم بعرض الصفحة على جهازه.. إن هذا التقسيم يقدم للعميل العديد من المزايا: فهو لا يحتاج لشراء التطبيق، ولا يحتاج لإعداده على جهـازه، ولا يستهلك مساحة تخزين لديه، ولا يقلق بخصوص تحديثه والإصدارات الجديدة منه.
- يتم ترجمة Compile كود التطبيق في أول مرة يتم تنفيذه فيها.. هذا هو السبب في البطء النسبي للتنفيذ فـي أول مرة.. بعد ذلك لا يتم ترجمة الكود طالما أن التطبيق يعمل.
- يقوم C# بإنشاء خلية Class لتتفاعل مع متصفّح الإنترنت.. هذه الخلية تنشئ وتترجم في وقت التنفيذ.
- تقوم هذه الخلية بقراءة قيم المتغيرات المرسلة من المتصفّح، وتعالج الاستعلامات المرسلة، ثم تُنتج كود HTML وترسله إلى العميل.. معنى هذا أنّك تستطيع تغيير قيم الأدوات على صفحات الإنترنـت، كمـا تـستطيع إنـشاء صفحات جديدة بها القيم الجديدة للأدوات وإرسالها للعميل.
- ويجب أن تضع في اعتبارك أن الكود يتم تنفيذه على الخادم Server، بمعنى أنّه لا يرى القـيم الموجـودة فـي الأدوات إلا حينما يرسلها له المستخدم (بضغط زر ما).. إذن فالتطبيق لا يعمل على الخادم طوال الوقت، وإنّمـا يتم تشغيل الكود اللازم عندما يطلب العميل ذلك.. وتسمى عملية إرسال كلّ قيم الأدوات إلـى الخـادم بإرجـاع البيانات Postback.

بيئة الدوت نت

هي بيئةُ تطويرٍ متكاملةٌ (IDE) Integrated Development Environment، لبنـاءِ واختبـارِ وتـصحيحِ البرامجِ المتنوعة: تطبيقات ويندوز Windows Applications، تطبيقات الإنترنت Web Applications، الخلايا Classes، وأدوات التحكم الخاصة Custom Controls، وهي تمنحُك العديد مـن الأدواتِ المرئيـةِ Visual Tools لإنشاء واجهة التطبيق Application Interface بشكلٍ شبه آلي، ولتـسريع وتـسهيل أداءِ العديدِ من المهام في التصميمِ والبرمجةِ.

ASP.NTهو جزء من هيكل ال NET.لمايكروسوفت

لكن مما يتكون هيكل ال NET.؟

لدى هيكل ال NET.عنصران :

مكتبة فصول شاسعة و بيئة تشغيل. (run-time)

تصنيف مكتبات هيكل الNET.

تصنيف مكتبة هيكل ال NET.يحتوي على ما يزيد عن ٣٤٠٠ طبقة يمكن أن تستخدمها في طلباتك . تمثّل بعض من هذه الطّبقات أنواع البيانات الأساسية مثل النصوص (string) والمصفوفات (array) . أغلبية الطّبقات مخصصة لمهام متخصصة أكثر مثل العمل بنظام الملفاتّ و توليد وثائق. XML

مجموعات المكتبات(Namespaces)

الطّبقات في هيكل ال NET.لا تُخْلَط ببساطة معًا . الطّبقات توزع على شكل مجموعات هرمية تسمى . (*namespaces*)على سبيل المثال ,كلّ الطّبقات التي تعمل مع نظام الملفاتّ تُجْمع في المجموعة .(System.IO)يمكن أن تشير إلى أي طبقة بشكل فريد في هيكل ال NET.بتزويد الصفحة بإسم المجموعة (*namespace*)واسم الطّبقة (نوعها . (على سبيل المثال، للإشارة إلى طبقة الملفاتّ في مجموعة , (System.IO)يمكن أن تستخدم:

System.IO.File

مجموعة واحدة من (*namespace*) مهمة خصوصًا لبرمجة صفحات . ASP.NET كلّ الطبقات التي تتعلق بهيكل ASP.NETيمكن أن تُوجد تحت المجموعة . (System.Web) كلّ تحكم أشكال الفورم في HTML والتي تسمى (HTML controls)موجودة في مجموعة System.Web.UI.HTMLControls و كلّ تحكّم أشكال الفورم في ASP.NET والتي تسمى (Web controls) موجودة في مجموعة. System.Web.UI.WebControls

ملفات المكتبات(Assemblies)

الطّبقات في هيكل ال NET.تجمع في ملفات تسمى . (Assemblies) من المهم عدم الخلط بين الملفات (Assemblies)والمجموعات . (*namespace*) الملف (assembly) هو الملفّ أو الملفات الفعلية التي تحتوي على الكود للطّبقات . على سبيل المثال، كلّ الطّبقات في المجموعة (System.IO) تقع في الملف . Mscorlib.dll ملفّ Mscorlib.dll هو الملفّ الفعلي الذي ستجده على القرص الصلب في كمبيوترك.

من ناحية أخرى، لاترتبط المجموعة (*namespace*) مباشرةً بالطّريقة التي وزعت بها الطّبقات التي يحتويها في الملفات . (Assemblies) يمكن أن تحتوي المجموعة (*namespace*) الواحدة عدة ملفات () وكذلك يمكن أن يحتوي الملف الواحد (assembly) على أكواد عدة مجموعات . (*namespace*) وبمعنى آخر فأن المجموعة هي تجميع منطقي (logical) للطبقات بينما الملف هو تجميع حسي (physical) لها.

لغة وقت تشغيل (Runtime) مشتركة في ال.NET

العنصر الثّاني لهيكل ال.NET بيئة وقت تشغيل (run-time) تسمى لغة وقت تشغيل (Runtime) مشتركة. عندما تنشيء صفحة ASP.NET، الصفحة تُجْمع (compile) تلقائيا في طبقة عند طلبها للمرة الأولى . عندما تُجْمع

(compile) الصفحة، لا تجمع مباشرةً إلى لغة الآلة الأصلية . بدلاً من ذلك، الصفحة تُجْمع الى شيئ ما يسمى اللّغة المتوسطة لمايكروسوفت Microsoft Intermediate Language وتسمى إختصارا (MSIL) .
MSIL متشابهة مع كود لغة الملفات (assembly) ، باستثناء أن لديه سمات البرمجة بالاهداف (-object oriented) و صُمم ليكون غير معتمد على نوع الجهاز.
بصرف النّظر عن اللّغة التي تستخدمها لكتابة صفحة ASP.NET، الصفحة تُجْمع (compile) الى نفس كود MSIL . ومن ثم كود MSIL يحول الى لغة الآلة بواسطة

(الترجمة في لّحظة) Just-in-Time أو (JIT). وهى تقنية تقوم بترجمة البرنامج عند تنفيذة حيث ينتج افضل شيفرة تتناسب مع الجهاز الذى سيعمل علية البرنامج في الحقيقة أن تحويل صفحة ASP.NET الى كود MSIL قبل التّنفيذ لة بعض المميزات المهمة .
أولا، تجعل إختيار لغة برمجة صفحة ASP.NET مجرد تفضيل شخصي . على سبيل المثال، يمكن أن تكتب صفحة ASP.NET بإستخدام لغة Basic Visual التي تنفّذ بسرعة مثل أي صفحة مكتوبة بلغة C# أو C++ .
بصرف النّظر التي تستخدمها لكتابة صفحة ASP.NET ، الصفحة تُجْمع (compile) في نفس شفرة MSIL .
أيضًا، بما أن كلّ لغات .NET تنفذ أو تجمع (compile) الى نفس كود MSIL ، إذن جميع لغات .NET تعمل مع بعض جيدا. على سبيل المثال، يمكن أن تنادي العنصر الذي كتبتة بإستخدام Visual Basic في صفحة ASP.NET مكتوبة بلغة C#

مقارنة بين ASP.NET وASP

ASP.NETمطور من . ASP لكن ASP.NET ,مطور ومتغير بشكل كبير عن ASP وكأن التقنيتين لاتتعلقان ببعضهما الا بشكل يسير .
بينما صفحات ASP التقليدية تكتب بسكريبتات بعض اللّغات مثل VBScript و ,JScript صفحات ASP.NET تكتب بلغات البرمجة الكاملة مثل لغة Visual Basic و .C# و بينما هناك خمسة عناصر (objects) عادية متاحة في ASP وهي ,(Request, Response, Application, Session, and Server objects) هناك ما يزيد عن ٣،٤٠٠ object عادي في هيكل ال .NETالذي يمكن أن تستخدمه في سيرفر. ASP.NET
من المدهش، على اعتبار كلّ القوة و المرونة جمعت في ASP.NET, ASP.NET أسهل كثيرًا للاستخدام من .ASPبإضافة تحكم سيرفر قليل إلى صفحة، يمكن أن تبني صفحة متطورة في دقائق والتي ستأخذ الأيام لتّطويرها ب. ASP

لغات ASP.NET

يمكن أن تكتب صفحة ASP.NET باستخدام أي لغة من اللغات المتوافقة مع الهيكل . NET. تتضمن هذه اللّغات لغة ,Visual Basic, C#لغة مطورة من سي ++، و .JScript.NET هذا يعني أنّك يمكن أن تطور معرفتك بإحدى هذة اللغات لبدء بناء صفحات ASP.NET بسرعة.

أيضًا هناك عدد من الشركات التي طورت لّغات إضافية للعمل بهيكل الـ . NET.على سبيل المثال، إذا فضلت، يمكن أن تكتب صفحات ASP.NET مع COBOL أو PERL.

تحكم سيرفرASP.NET

لمدة عدة سنوات ,يتمتع مبرمجو لغة Visual Baisc بالقدرة على إنشاء أشكال معقّدة بالتحكم مثل TextBox و ListBox. معASP.NET, يمكن أن تعمل نفس الشّيء مع صفحات. Web Form

يتضمن هيكل ASP.NET أربعة مجموعات للتحكم العادي:

١- تحكم واجهة المستخدم الأساسية مثل. TextBox, RadioButton, ListBox, and Button

٢- تحكّم الإجازة لإجازة معلومات الفورم . تتضمن مجموعة التحكم هذه فحص لحقول الفورم المطلوب إدخالها وفحص نوع البيانات التي أدخلت في الحقل وهل هي مناسبة أم لا.

٣- تحكم البيانات لعرض قاعدة البيانات و أنواع البيانات الأخرى . على سبيل المثال، يمكن أن تستخدم تحكم DataGridلعرض محتويات جدول قاعدة بيانات بسهولة.

٤- السيطرة الغنية لعرض عناصر واجهة المستخدم المعقّدة مثل التقويم أو إدارة البانرات الإعلانية.

من خلال ,Visual Studio.NET وكذلك برنامج ASP.NET Web Matrix المختص بتحرير هذه اللغة يمكن أن تسحب عناصر التحكم لتبني صفحات ويب معقدة . أو إذا فضلت، يمكن أن تنشيء صفحات ASP.NET من خلال برنامج المفكرة وتضع فيها أي من عناصر التحكم السابقة بكتابتها يدويا.

تحميلASP.NET

لا تحتاج إلى شراء أي شيئ لبدء البرمجة مع .ASP.NET يمكن أن تحمل ASP.NET بهيكل ال NET.من موقع هذه اللغة الرسمي:

تحميل ASP.NET الإصدار ١،٠

تحميلASP.NET Web Matrix

ASP.NETمتوافقة مع نظام التشغيل (Windows 2000 Server و (Professional وكذلك مع Windows XP.

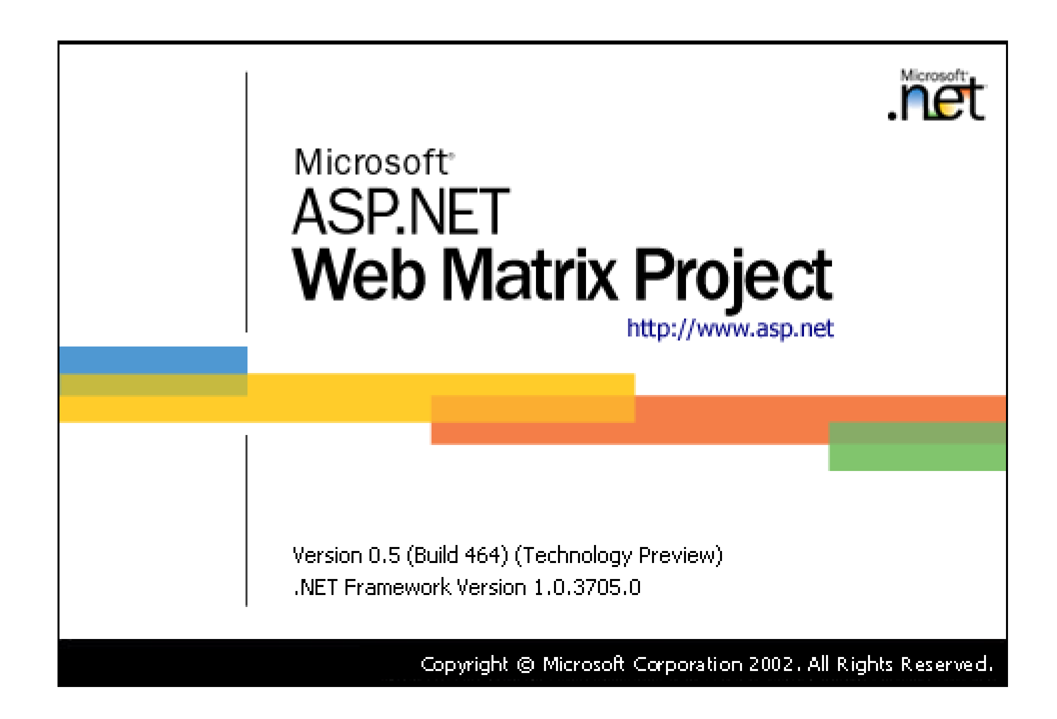
الباب الاول

التعرف على برنامج Microsoft ASP.NET Web Matrix

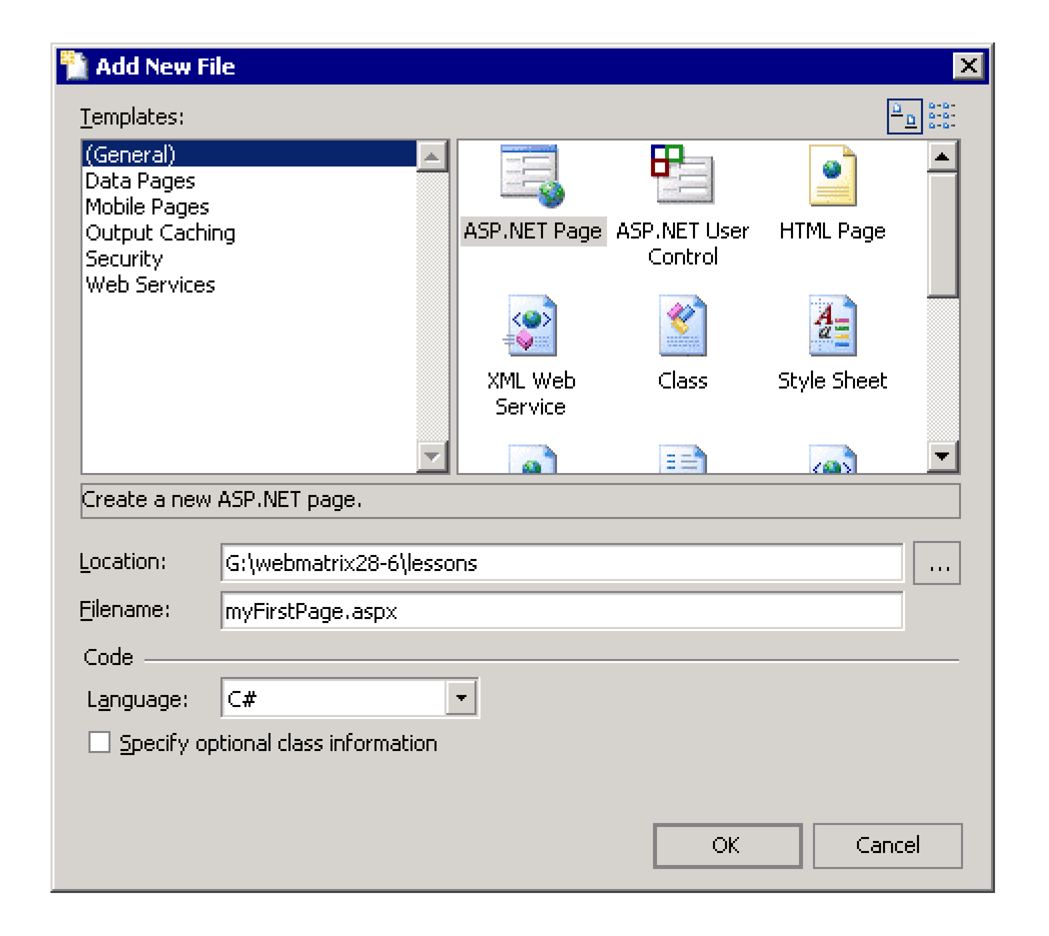
هذا البرنامج MS Web Matrix هو إصدارة من شركة Microsoft للعمل كمحرر لتقنية الدوت نت و يجب من أجل تثبيته قبلا تثبيت ال Dot Net Framework الموجودة ضمن أدوات ال SDK التي تقدمها ميكروسوفت للمبرمجين من خلال موقعها على الشبكة أو إسطوانات مدمجة و يمكن الحصول على محرر ال MS Webmatrix من الموقع الرسمي لشركة ميكروسوفت /http://www.microsoft.com

أو من موقع www.asp.net

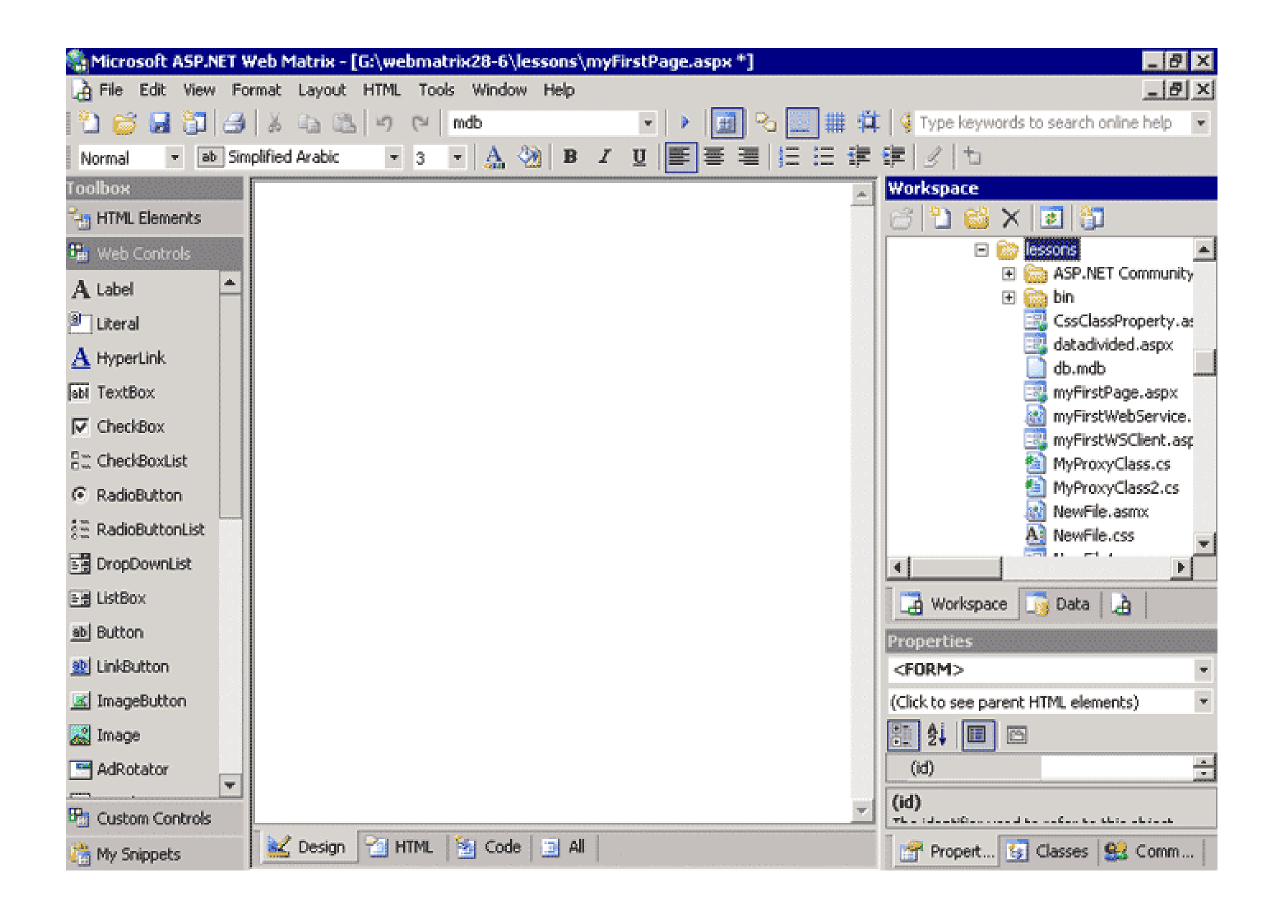
و هذه صورة المحرر:



عند تشغيل المحرر سوف تظهر لنا شاشة كالتالي:



يوجد بالأسفل مربع نص مكتوب بجانبه .Location سنقوم بتغيير المسار و لنجعله-G:\webmatrix28
6\lessonsو من ثم نغير اسم الملف بجانب ال File Name إلى myFirstPage.aspx أو أي إسم نريده و
لننتبه أن الإمتداد للملف يجب أن يكون aspx ثم نختار اللغة التي سنبرمج بها وأنا هنا إخترت الC# ثم انقر على
ok ثم Ctrl+s لحفظ الصفحة



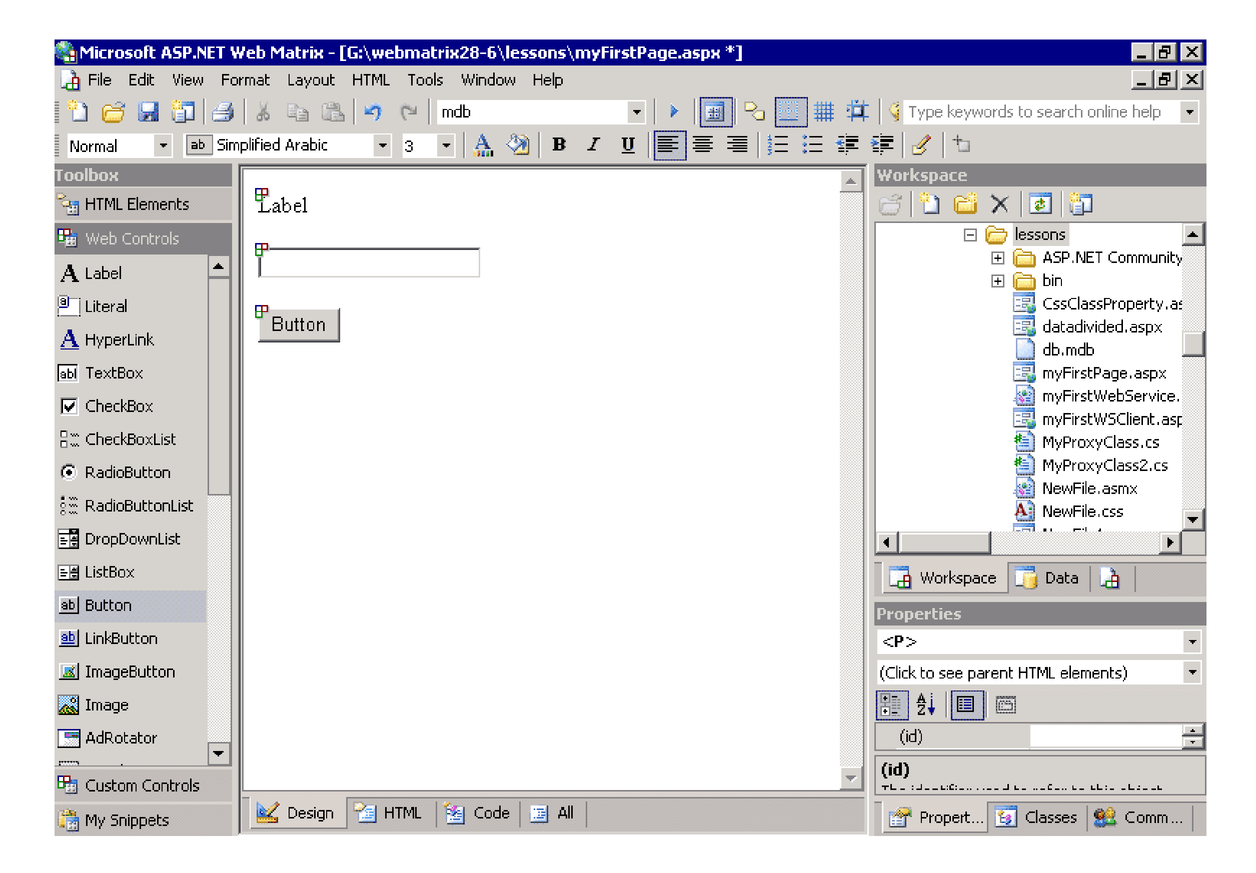
سوف تظهر لنا واجهة للعمل فارغة و بها شريط بالأسفل كالموضح بالصورة هنا



تظهر أمامنا واجهة التصميم Design و منها يمكن تصميم شكل النموذج بطريقة سهلة من خلال استخدام الادوات الموجودة على الجهة اليسار من البرنامج و ككل برامج ميكروسوفت سوف نستخدم خاصية السحب و الافلات فى تصميم صفحات aspx و لنبدأ بعمل صفحة بسيطة تظهر بها عبارة hello world و لكن سوف نستخدم ثلاث أدوات Controls .

١ - اذهب لصندوق الادوات ToolBox و انقر على التبويب Web Controls و اسحب و

Label control, a TextBox control and a Button control و اتركهم على النموذج و لا تنسى ان تفصل بينهم بالضغط على Enter من لوحة المفاتيح لاحظ الشكل



انقر على تبويب Html فى أسفل الصفحة لكى ترى كود Html الذى قام بصنعة البرنامج

```
1.   <html>
2.   <head>
3.   </head>
4.   <body>
5.       <form runat="server">
6.           <p>
7.               <asp:Label id="Label1" runat="server">Label
8.               </asp:Label>
9.           </p>
10.          <p>
11.              <asp:TextBox id="TextBox1" runat="server">
12.              </asp:TextBox>
13.          </p>
14.          <p>
15.              <asp:Button id="Button1" onclick="Button1_Click" runat="server"
     Text="Button">
16.              </asp:Button>
17.          </p>
18.      </form>
19.  </body>
20.  </html>
```

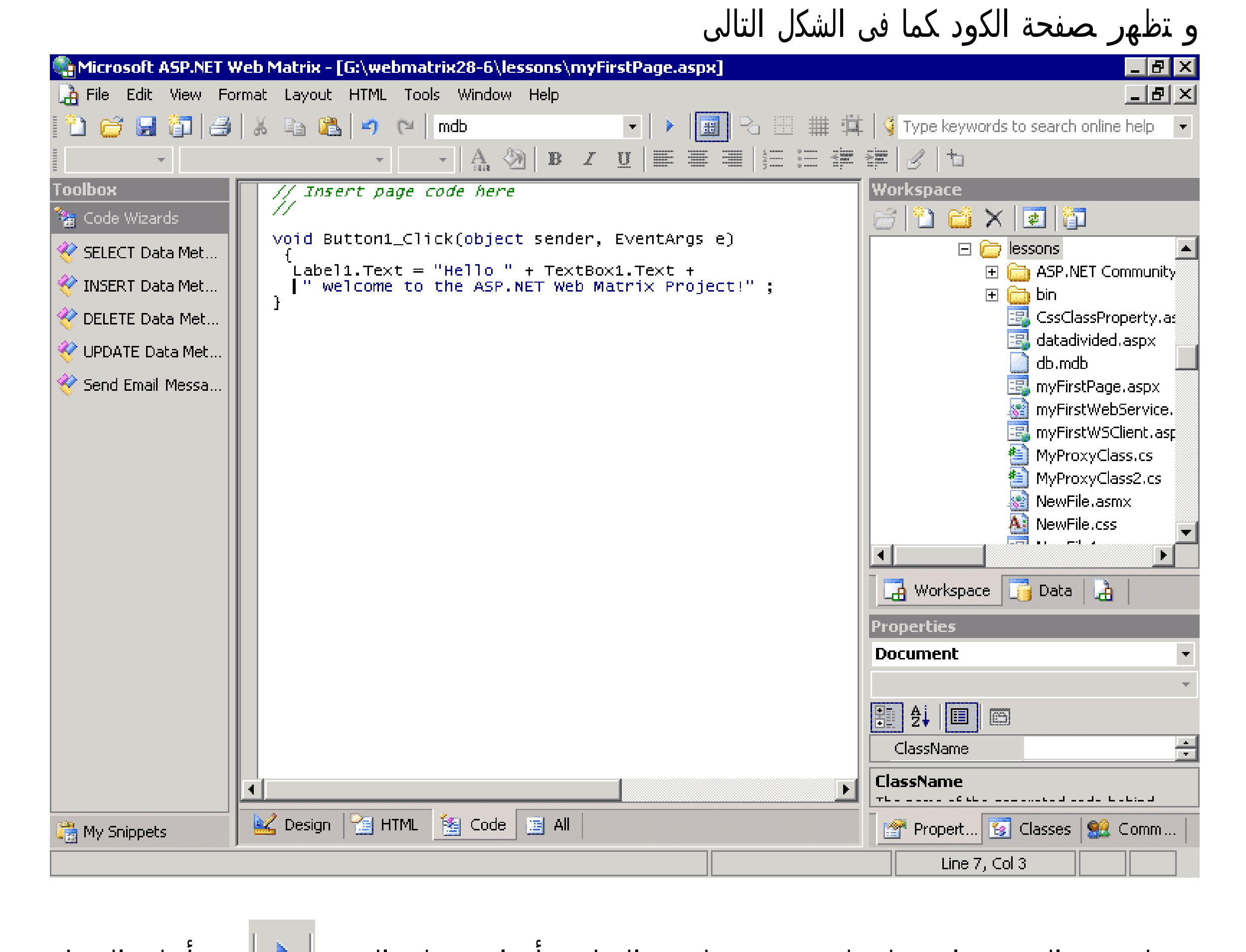
٢- قم بالتغيير للوضع Design و انقر على الزر Button1 فتظهر شاشة كتابة الكود Code و قم بأضافة الكود التالى

Label1.Text = "Hello " + TextBox1.Text + " welcome to the ASP.NET Web Matrix Project! ";

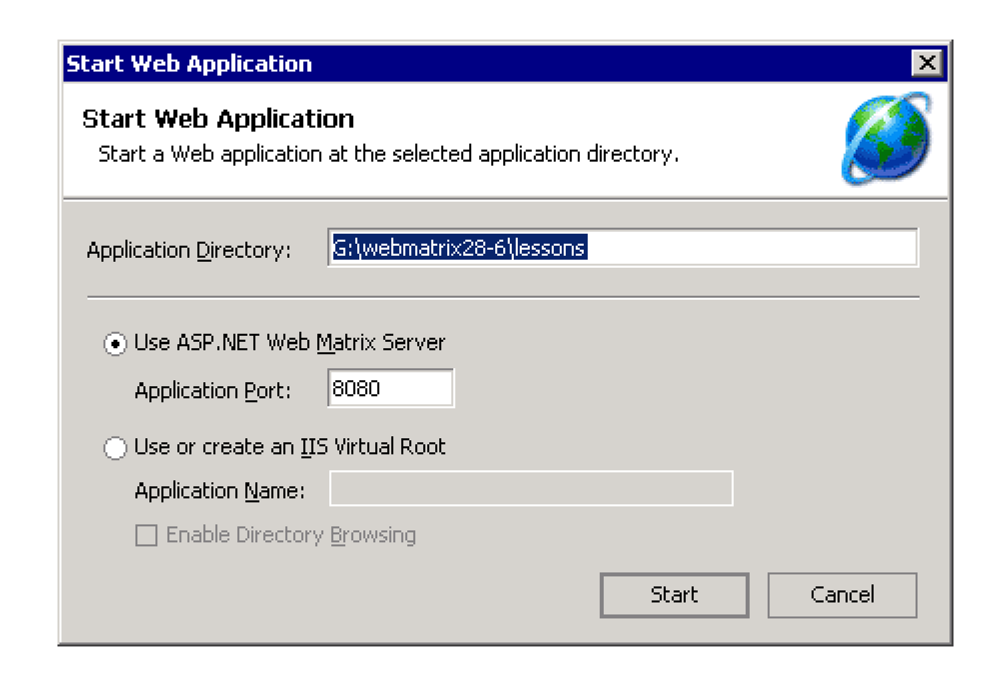
ثم أضغط مفتاحىctrl+s معا لحفظ الصفحة أو أنقر على أيقونة الديسك

تفسير الكود السابق هو عرض كلمة Hello ثم أى كلمة نقوم بكتابتها فى مربع النص TextBox1 بجانب نص ثابت هوwelcome to the ASP.NET Web Matrix Project! و لاحظ ان النصوص الثابتة قد وضعت بين " " و استخدمنا علامة + للربط

و تظهر صفحة الكود كما فى الشكل التالى



٣- لتجربة الصفحة اضغط على F5 من لوحة المفاتيح أو انقر على السهم فى أعلى البرنامج فيظهر مربع حوار يسأل على رقم المنفذ Port في الجهاز و الذي من خلاله سوف يتم تنفيذ الكود و عرضه . اتركها كما هي ٨٠٨٠ و إضغط على Start



فيتم تنفيذ الصفحة من خلال السيرفر المدمج مع البرنامج و يفتح المتصفح صفحة جديدة كالتالى

http://localhost:8080/myFirstPage.aspx - Microsoft Internet Explorer

Label

Button

بعد ذلك قم بكتابة أسمك مثلا فى مربع النص و انقر على الزر Button فيظهر التالى

http://localhost:8080/myFirstPage.aspx - Microsoft Internet Explorer

Hello مايكل نبيل welcome to the ASP.NET Web Matrix Project!

مايكل نبيل

Button

لاحظ ظهور الشكل التالى عند شريط المهام فى جهازك

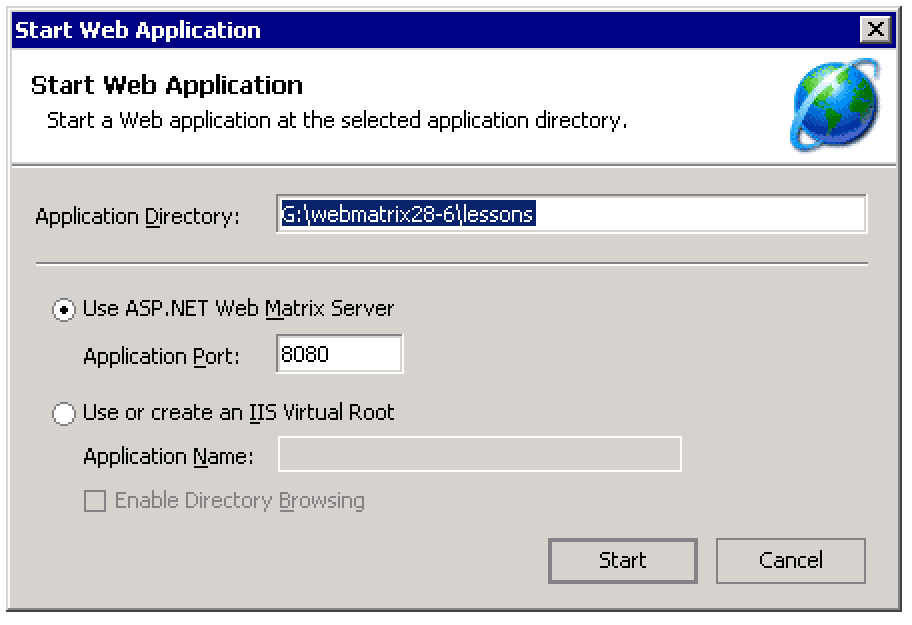


و هذا يعنى أن السيرفر الخاص ببرنامج Web Matrix يعمل جيدا و اذا أرت أيقافة عن العمل أضغط على الأيقونة الخاصة بة بزر الماوس الايمن لتظهر لك القائمة الفرعية التالية



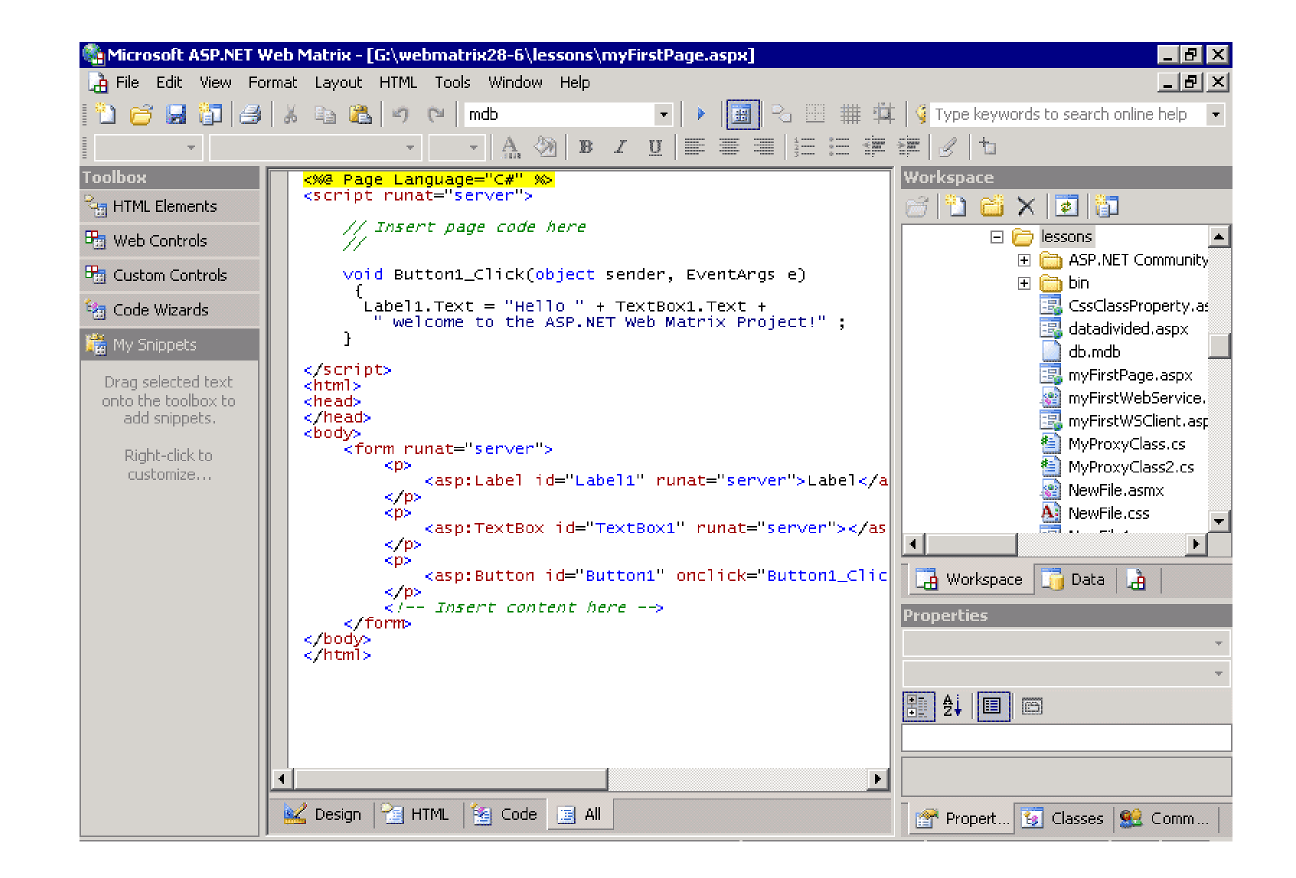
و اختر من القائمة Stop أو لآعادة التشغيل Restart .

أما أذا كنت تريد استخدام السيرفر المدمج مع Windows2000 فأختر من الشكل



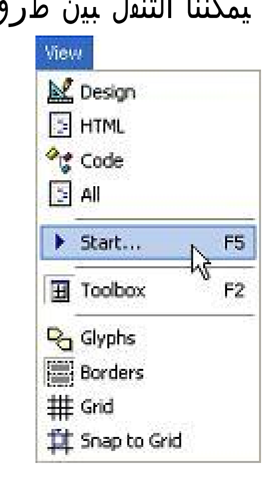
Use or create an IIS Virtal Root و اكتب فى خانة Application Name الاسم الذى تريدة للمجلد و ليكن مثلا Lessons .

٤- الان دعنا نرى تبويب All بالنقر علية أسفل الصفحة فيظهر لنا الشكل التالى



أن أختيار All تظهر لي كل شيء مكتوب سواء HTML أو الأكواد المكتوبة برمجيا نرى فيها باللون الأصفر بالأعلى وجود موجه Directive يحدد اللغة التي سنكتب بها و هنا مكتوبة C# و بالأسفل سنرى وسمين Tags رئيسيين واحد لكتابة الأكواد scripts و يسمى **كود التوضيح (code declaration block) و الثاني لعناصرالHTML و يسمى** قسم كود التجميع (code render block)

يمكننا التنقل بين طرق العرض داخل البرنامج من قائمة View و ايضا تشغيل الصفحة   كما بالشكل :



الان سوف نذهب للوسم Script و نكتب به كودا بسيطا للغاية يظهر مسار الصفحة التي نحن بها

```
void Page_Load()
{

object Path=Request.ServerVariables[30  ];
Response.Write("this page Physical path:|<br>"+Path);
```

```
}
```

لنجرب الصفحة اضغط F5 ثم أضغط على الزر فيظهر الصفحة التالية

this page Physical path:|
G:\webmatrix28-6\lessons\myFirstPage.aspx

Hello مايكل نبيل welcome to the ASP.NET Web Matrix Project!

مايكل نبيل

Button

ان الغرض من المثال ليس الكود البرمجى و لكن أيضاح انة يمكنك كتابة الكود فى شاشة All
و ذلك يدفعنا للتحدث عن موضوع هام هو Code Render Blocks
ما هو؟؟ هو أمكانية كتابة الكود بين وسوم ال Html بشرط أن يكون الكود مكتوب باللغة المستخدمة فى الصفحة أساسا و أن يكون الكود سبق ان تم تعريفة داخل الوسم <Script>
و لتستخدمه داخل أكواد الHtml يجب ان تضعة بين <%   %> و المثال التالى يوضح الفكرة :

١- افتح صفحة myFirstPage.aspx و انقر على تبويب All و عرف متغير من النوع String و اعطى له الاسم strMessage

أضف السطر التالى للكود قبل void Page_Load()

```
String strMessage;
```

٢- و عند الجزء

```
{

object Path=Request.ServerVariables[30  ];
Response.Write("this page Physical path:|<br>"+Path);
```

أضف السطر التالى //

```
strMessage= "Hallo and welcome to our web site" ;

}
```

٣-و أضف الكود التالى بعد الوسم <body>

```
<h3> <%=strMessage%></h3>
<p> our web site offers many things <p>
<%"p><%="we hope you enjoy with us>
```

٤-و الان لنرى الكود ككل

```
<% @ Page Language= "C# "%>
<  script runat=" server">
```

```
//Insert page code here
// أضف السطر التالى
String strMessage;
void Page_Load()

{
object Path=Request.ServerVariables[30 ];
                                        +Path);"Response.Write(“this page Physical path<br>

                                                ;"Hallo and welcome to our web site"strMessage =
}

void Button1_Click(object sender, EventArgs e)
{
Label1.Text = “Hello “ + TextBox1.Text     +
"!welcome to the ASP.NET Web Matrix Project "     ;
}

<script/>
<html>
<head>
<head/>
<body>
<h3> <%=strMessage%></h3>
<p> our web site offers many things </p>
<p><%=we hope you enjoy with us%>
<p><% ="  we hope you enjoy with us"%>

    <form runat="server">
            <p>
                <asp:Label id="Label1" runat="server">Label
                </asp:Label>
            </p>
            <p>
                <asp:TextBox id="TextBox1" runat="server">
                </asp:TextBox>
            </p>
            <p>
        <asp:Button        id="Button1"        onclick="Button1_Click"        runat="server"
    Text="Button">
                </asp:Button>
            </p>
```

```
</form>

<--! Insert content here-->
<body/>
<html/>
```

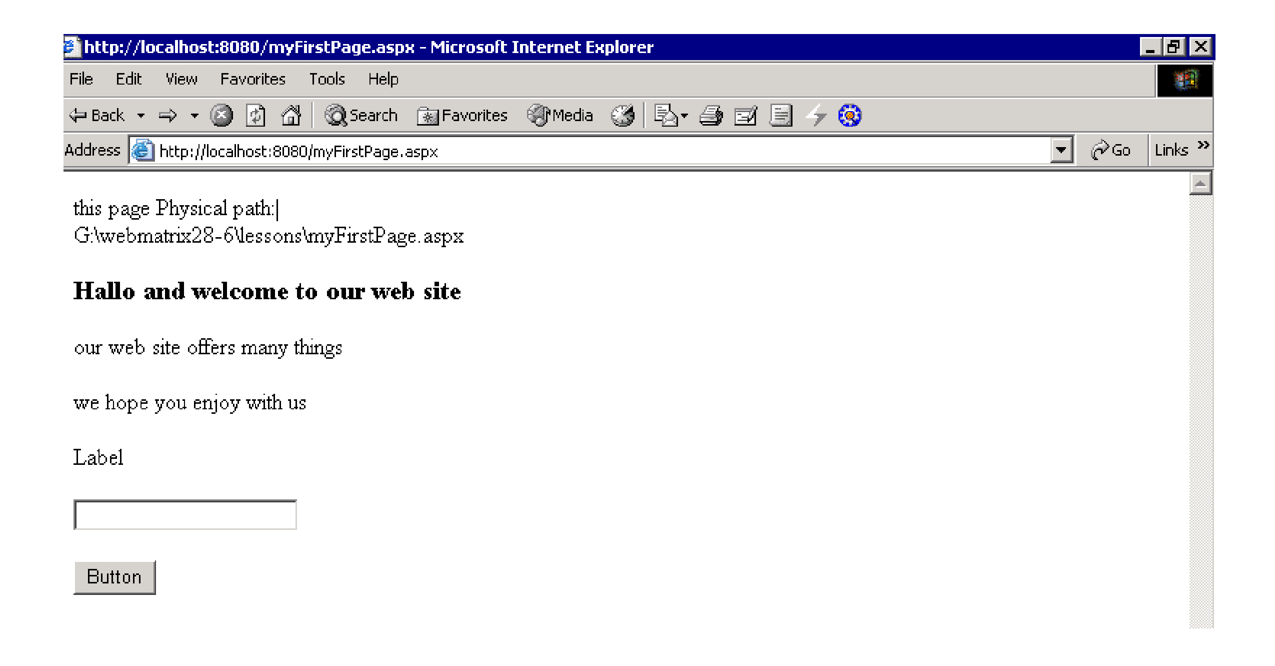
اعتقد أنك لاحظت سطر غريب بين الكود هو

أضف السطر التالى //

أنة تعليق على الكود يمكنك أستخدام(//) كطريقة لتجنب تنفيذ سطر معين من الكود او لكتابة ملاحظات معينة و لكن أذا كنت ترغب بكتابة عدة أسطر استخدم

/*............ أكتب كل ما تريد بين العلامتين*/

٥- الان لنقم بتجربة الصفحة بالضغط على F5 و الزر فيظهر الشكل التالى :



كما سبق و قلت ان الغرض من المثال السابق هو توضيح اين يمكن كتابة الكود و ذلك لكى نبدأ تعلم أساسيات لغة الC# و تطبيق الامثلة من خلال البرنامج و الان لنبدأ الدروس الاساسية لتعلم لغة ال C# :

اولا :المتغيرات Variables

لماذا نعرف المتغير؟:

في معظم لغاتِ البرمجة، يجبُ تعريفُ المتغيراتِ أولا قبلَ استخدامها.. إن هذا يجعل الأمر أيسر بالنسبة لمترجم الكود Compiler، ففي كلّ مرة يصادف المترجم متغيرا، عليه أن يُنشئه في الذاكرة، ونتيجة لاعتباراتٍ في تنظيم الذاكرة، فإن مثل هذه العملية تستهلك بعض الوقت، مما يُبطئ البرنامج.. ولكن لو كان المترجم يعرف كل متغيرات البرنامج وأنواعها سلفا قبل أن يبدأ ترجمة البرنامج، ففي هذه الحالة سيتحسن الأداء لأقصى درجة.

لقد كانت من أشهر سمات VB، عدم إرغامه للمبرمج على تعريف كلّ المتغيرات.. لقد صارت هذه السمة منتقدةً الآن بشدةٍ، ليس فقط للأسباب المتعلّقة بسرعة الترجمة وكفاءة الأداء، ولكن أيضا لأن تعريف المتغير يُمكّن المترجم من اصطياد أخطاء كثيرةٍ، سواء في وقت التصميم Design Time أو وقت الترجمة Compile Time، بدلا من أن تُفاجئك في وقت التشغيل Runtime.

فمثلا: عندما تعرف متغيرا من النوع "تاريخ" Date</span:، لا يمكن أن يسمح لك المترجم بوضع عدد صحيح Integer فيه.. أيضا لن يسمح لك المترجم باستخدام خاصية "شهر" Month الخاصة بالمتغيرات من النوع "تاريخ" Date، مع متغير من النوع "عدد صحيح" Integer.

إن تعريف المتغيرات يمكّن المترجم من اصطياد مثل هذين النوعين من الأخطاء، أثناء كتابتك للكود، فلا تكاد تترك السطر الذي كتبتها به، حتّى يضع تحته خطًا متعرجا، لو حلّقت فوقه بالفأرة، فسيظهر لك تلميح على الشاشة يصف الخطأ الذي ارتكبته.

كما في أي لغة برمجة، تقوم المتغيرات بتخزين القيِم أثناء تنفيذ البرنامج.. وطبعًا سُميت متغيراتٍ، لأنّك تستطيع تغيير قيمها في أي لحظةٍ أثناء تنفيذ البرنامج.

وللمتغير اسمٌ وقيمة.. فمثلا: المتغير "اسم المستخدم" UserName يمكن أن تُوضع به القيمة "ماهر".. والمتغير "الخصم" Discount يمكن أن تُوضع به القيمة ٠,٣٥.

تلاحُظ هنا أن القيمتين "ماهر" و ٠,٣٥ مختلفتان، فالأولى نص String لهذا تم وضعها بين علامتَيْ تنصيص، بينما الثانية قيمة رقمية Numeric Value.

وكما ذكرنا سابقا، فإن المتغيرات في C#.NET ليست مجرد أسماءٍ أو مخازن للقيم.. إنّها كذلك كِيانات ذكيةٌ لتخزين وإجراء العمليات على القيم.. باختصار: إنّها كائنات Objects، لها وسائلها وخصائصها الخاصة بها.

أنواع المتغيرات الرقمية:

إن للمتغيرات الرقمية أنواعًا عديدة، تبعًا لحجم العدد ودقّته العشرية.. وعليك أنت أن تحدد النوع الذي يناسبك، واضعًا في الاعتبار أن الأعداد ذات الدقّة العشرية الأكبر، تكون العمليات عليها أبطأ من الأعداد الصحيحة والأعداد ذات الدقّة العشرية الأقلّ.

وفي هذا الجدول، الأنواع التي تستطيع استخدامها لتخزين القيم الرقمية في #C:

النوع	حجمه بوحدات الذاكرة Bytes	الأعداد التي يقبلها
وحدة ذاكرة Byte	١	من ٠ إلى ٢٥٥.
وحدة ذاكرة ذات إشارة SByte	١	من -١٢٨ إلى ١٢٧.
قصير Short Int16	٢	من –٣٢٧٦٨ إلى ٣٢٧٦٧
صحيح Integer Int32	٤	من -٢١٤٧٤٨٣٦٤٨ إلى ٢١٤٧٤٨٣٦٤٧
طويل Long Int64	٨	يقبل أرقاما كبيرة جدا.
مفرد Single	٤	للأعداد ذات الدقّة العشرية العادية.. وهو يمثّل الأعداد السالبة: من 3.402823E38– إلى –1.401298E–45 ، والأعداد الموجبة من 1.401298E–45 إلى 3.402823E38. وفي هذا النوع، لا يمكن تمثيل الصفر بدقّة، فهو يمثّل دائما بقيمة صغيرة جدا جدا، ولكنها أبدا ليست صفرا!
مزدوج Double	٨	للأعداد ذات الدقّة العشرية الفائقة، وهو يستخدم في الحسابات العلمية.
عشري Decimal	١٦	لتمثيل الأعداد الصحيحة والعشرية، حيث يمكّنك من تحديد الدقّة العشرية التي تريد التعامل معها، من ٠ إلى ٢٨ خانة عشرية.

ملحوظة:

الرمز E في عدد مثل 1.401298 E –45 هو طريقة الكمبيوتر لتمثيل الأس العشري، أي أن هذا الرقم يساوي: ١,٤٠١٢٩ × ١٠ $^{-٤٥}$.

العمليات على الأرقام:

لن تكون هناك فائدة إذا كنت ستضع الأرقام في متغيرات، دون أن تستطيع أن تُجري عليها بعض العمليات الحسابية..
وفي الجدول التالي علامات العمليات الحسابية الأساسية:

+	علامة الجمع.
-	علامة الطرح.
*	علامة الضرب.
/	علامة القسمة.. ويمكن أن يكون الناتج عددا صحيحا أو به أرقام عشرية.. فمثلا: X = 7 / 2 ستعطي الناتج ٣,٥.
\	علامة القسمة أيضا، ولكن الناتج هو العدد الصحيح فقط.. فمثلا: X = 7 \ 2 ستعطي الناتج 3. ويمكن أداء نفس العملية باستخدام الدالة Int، لو شئت ألا ترتبك بين علامتي القسمة المتشابهتين، وذلك كالتالي: X = Int (7/2);
Mod	إحدى علامات القسمة أيضا، ولكنّها تعطي الباقي من القسمة فحسب.. فمثلا: X = 7 Mod 2 سيعطي الناتج ١، الذي هو عبارة عن باقي القسمة.
^	الأس.. فمثلا ٢ × ٢ × ٢ تُكتب رياضيا بالصيغة $٢^٣$، وتكتب في البرمجة كالتالي: 2 ^ 3 ولو أردت أن تعبر عن الجذر التكعيبي مثلا، فارفع العدد للأس (١ ÷ ٣) كالتالي: 2 ^ (1/3)

ويجب أن ألفت انتباهك إلى أهمية وضع الأقواس في العمليات المتداخلة، وذلك حتّى تضمن صحة إجراء العملية بالترتيب الذي تريدها به.. إن الترتيب الطبيعي الذي يجرى به العمليات الحسابية يسير تبعا للقواعد التالية:

- يتم تنفيذ ما بين الأقواس أولا.
- إذا لم تكن هناك أقواس يتم تنفيذ الأسس أولا.
- ثم يتم تنفيذ الضرب والقسمة.
- ثم بعد ذلك يتم تنفيذ الجمع والطرح.

لهذا فإن الصيغة:

8^(1/3)

تعطي الناتج ٢، وذلك لأن القوس ينفّذ أولا، فتصبح العملية هي الجذر التكعيبي للعدد ٨.. ولكن لو أزلت الأقواس كالتالي:

8^1/3

فإن الناتج سيكون ٢,٦٦٦٦٦٦، وذلك لأن الأس ينفّذ أولا، فتصبح العملية كالتالي:

8/3

فانتبه لهذا جيدا.

بقي شيءٌ هام.. ماذا لو أردت أن نزيد قيمة متغير بمقدار ١ مثلا؟

في هذه الحالة سنقوم بالتالي:

X = 5

Y = X + 1

X = Y ' **صارت قيمة المتغير ٦**

حيث اعتمدنا على متغير وسيط، جعلنا قيمته هي ناتج جمع المتغير الأصلي مع الواحد، ثم نقلنا قيمته إلـى المتغيـر الأصلي.

ولكن مثل هذه العملية تتكرر مرارا في البرمجة، حيث تحتاج مرارا لزيادة قيم المتغيرات أو إنقاصها، أو ضربها فـي رقم... إلخ.

فلو كان على المبرمج أن يكتب هذه الخطوات في كلّ مرة، لصارت البرمجة جحيما لا يُطاق!

لا تقلق.. يمكنك أداء هذه العملية في سطر واحد مباشرةً كالتالي:

X = X + 1

لأول وهلةٍ ستبدو لك الصيغة غريبة، ولكن حاول أن تقرأها كالتالي: قيمة X الجديدة تساوي قيمته القديمة + ١.

ولا يشترط أن أجمع على المتغير الرقم ١ فحسب، فهذه العمليات أيضا مباحة:

X = X + 13

X = X + X

Y = 5

X = X + Y

ولا يقتصر الأمر على الجمع فحسب، بل يمتد إلى باقي العمليات الحسابية:

X = X – 4 ' **إنقاص المتغير بمقدار ٤**

X = X * 2 ' **ضرب المتغير في ٢**

X = X / 9 ' **قسمة المتغير على ٩**

X = X ^ 3 ' **رفع المتغير للأس ٣**

هناك تسهيلا إضافيا تقدمه لك C#.Net، عن طريق استخدام الرموز += و -= ، *=، /=، ^=.. والجـدول التـالي يريك كيفية استخدامها:

الطريقة المختصرة المكافئة	الطريقة التقليدية
X + = 1	**X = X + 1**
X += Y	**X = X + Y**
X -= 4	**X = X – 4**
X *= 2	**X = X * 2**
X /= 9	**X = X / 9**
X ^= 3	**X = X ^ 3**

وأنت حر في اختيار الصيغة التي تريحك.

ملحوظة:

عند استخدام الأس ^ في أي عملية، تعتبر اللغة أن الناتج سيكون Double، وذلك لأن هذه العملية في الغالب تُعيد أرقاما كبيرة

المتغيرات المنطقية Boolean Variables:

المتغيرات المنطقية تخزن واحدةً فقط من القيمتين: "صواب" True و"خطأ" False، وهي في الأساس أعداد صحيحة، فالقيمة "صواب" تعادل -١، والقيمة "خطأ" تعادل صفرا.. وفي الواقع، أي قيمة غير صفرية، تعتبر True.

ويمكنك تعريف المتغيرات من هذا النوع، بجملة كالتالية:

```
Boolean  Failure
```

ولو لم تحدد قيمةً ابتدائية، فستكون قيمة المتغير المبدئية False.

ولكن كيف نضع القيم في المتغير Failure؟.. إن كلّ التعبيرات التالية متاحة:

```
Boolean  Failure  = True;
Failure = False;
' دالة تعيد قيمة منطقية Failure = CheckFailure()
```

ولا يتوقّف الأمر عند هذا الحد، بل يمكن أن تضع في المتغير Failure نتيجة أي عملية مقارنة

تعريف المتغيرات النّصية:

يتم تعريفها كالتالي:

```
String SomeText ;
```

الآن يصبح بإمكانك أن تضع أي نص، مها كانت مكوناته (حروف، رموز، أرقام، علامات تنسيق... إلخ)، ومهما كان طوله (يمكن أن يصل إلى ٢ جيجا بايت، أي ٢ مليار حرف!).

وهذه الجمل تُريك كيف تستخدم المتغير النصي:

```
String AString ;
AString = "اكتب ما تريد في هذا المتغير";
' متغير نصي فارغ AString = "";
AString = "ولكن بإمكانك أن تكتب بهذا المتغير نصا جديدا أطول من القديم";
'هذا نص يحتوي على قيمة رقمية، ويمكن استخدام دوال التحويل
' للتعامل معها كرقم وليس كنص
AString = "٢٥٠٠٠";
```

طبعا لاحظت أن النص يوضع بين علامتي تنصيص.. وهذا بالتأكيد سيدفعك للتساؤل: ماذا لو أردت أن أضع علامـة التنصيص نفسها في متغير؟

في هذه الحالة يجب أن تكتب أربع علامات تنصيص كالتالي:

```
String  AString = """";
```

وهناك حلّ آخر، هو استخدام الدالة Chr للتعبير عن علامة التنصيص كالتالي:

```
String AString  = Chr(34);
```

سؤال آخر سيراودك: ما الفارق بين التعبيرين التاليين:

```
Integer ANumber = 25000;
String AString = "25,000";
```

إن كلا المتغيرين يحمل قيمة مختلفة عن الآخر:

١- فالمتغير النصي AString يحتوي على ستة حروف، وهي "٢" و "٥" و "," و "٠" و "٠" و "٠"، بينما العـدد الصحيح ANumber يحتوي على رقم واحد هو ٢٥٠٠٠.

٢- بما أن كل حرف يتم تخزينه في وحدتي ذاكرة 2 Bytes، فإن المتغير النصي يخزن حروفه الـستّة فـي ١٢ وحدة ذاكرة، بينما يخزن العدد الصحيح العدد ٢٥٠٠٠ في ٤ وحدات فقط.

ولكن برغم هذه الخلافات، فإن بإمكانك أن تستخدم المتغير النصي AString في العمليات الحسابية، وتستخدم المتغير الرقمي ANumber في العمليات النّصية، حيث تقوم c# بالعمليات اللازمة للتحويل بين النوعين.

المتغيرات الزمنية Date Variables:

يتم تخزين متغيرات التاريخ والزمن بتنسيق خاص كأعداد مزدوجة، بحيث يمثّل العدد الصحيح التاريخ، ويمثّل الجزء العشري الزمن.

ويتم تعريف هذا النوع من المتغيرات والكتابة فيه بطرق كالتالية:

```
Date Expiration ;
Expiration = #01/01/2004#;
Expiration = #8/27/2001 6:29:11 PM#;
Expiration = "July 2, 2002";
Expiration = Now() ; دالة "الآن" تعطيك التاريخ والوقت الحاليين.
```

تلاحظ أنّنا نكتب التاريخ بطريقتين: إما كتاريخ رقمي مكتوب بين علامتي ##، أو كتاريخ نصي مكتوب بين علامتي تنصيص ""، حيث سيتم تحويله ضمنيا في الحالة الأخيرة، إلى تنسيق التاريخ والوقت المناسب.

كما أن بإمكانك استخدام الدوال الجاهزة التي يمنحها لك c# للتعامل مع التاريخ والوقت.. فمثلا: يمكنك استخدام الدالة "فرق التاريخ" DateDIff()، لحساب الفرق بين تاريخين مختلفين، وبالوحدة الزمنية التي تختارها (أيـام، سـاعات، ...).. إليك هذا المثال، لحساب عدد الأيام التي عشناها في هذه الألفية:

```
Long Days ;
Days = DateDIff(DateInterval.Day, #12/31/2004#, Now());
```

المتغيرات الكائنات Object:

الطريف في هذا النوع، أنّه يستطيع تخزين أي نوع من البيانات، سواء من الأنواع السابقة، أو مـن أي كـائن آخـر تعرفه اللغة، أو تعرفه أنت.

لكن لا تبدو الأمور بهذه البساطة، ففعلُ ذلك سيكون له آثار جانبية سيئة، فقبل أن يستخدم C# هذا المتغير الكـائن، لا بد له أن يعرف نوع البيانات التي به أولا، ليقوم بعمليات التحويل المناسبة، للتعامل مع هذا النوع من البيانات.. فمثلا، لو كان المتغير الكائن يحتوي على عدد صحيح، فعلى C# أن يحوله إلى نص قبل لصقه بنص آخـر.. مثـل هـذه التحويلات تمثّل عبئا على سرعة البرنامج.. لذا فعليك استخدام المتغير الكائن Object في حالات الضرورة فحسب.

تنسيق الأرقام Formatting Numbers:

الدالة "حول إلى نص" ToString هي إحدى وسائل كل أنواع البيانات، ما عدا البيانات النصية String بـ الطبع، وهي تقوم بتحويل الأرقام إلى نصوص، بأي تنسيق تريده.

وكما رأيت سابقا، يمكنك استدعاء هذه الدالة بدون أي معاملات، لتحويل العدد إلى نص كما هو بدون تنسيق.

ولكن بإمكانك أيضا أن ترسل لهذه الدالة معاملا، تبين به الصيغة التي يجب أن يُنسق العدد على شاكلتها.

وفي الجدول التالي، ستجد كل الرموز الجاهزة التي يمكن أن تستخدمها في التنسيق:

الرمز	المعنى	مثال	الناتج
C أو c	عملة Currency	12345.67.ToString("C");	سيتوقف الناتج على لغة نسخة الويندوز والمنطقة التي ضبطت عمله عليها، فلو كانت أمريكا، فسيكون الناتج $12,345.67، أما لو كانت مصر، فسيكون الناتج ١٢,٣٤٥،٦٧ ج.م.
E أو e	علمي Scientific	12345.67.ToString("E");	1.234567E+004
F أو f	مقربة لأقرب ٣ علامات عشرية Fixed-point	12345.6715.ToString("F");	12345.672
G أو g	عام General، وهي ترجع الرقم إما مقربا لأقرب ثلاث علامات عشرية، أو في تنسيق محدد.		
N أو n	تنسيق رقم Number	12345.67.ToString("N");	12,345.67
X أو x	عدد سداسي عشر HexaDecimal	250.ToString("X");	FA

ملاحظتان:

١- يمكن أن يُتبع رمز التنسيق برقم، ليدل علـى عـدد الخانـات العـشرية فـي الـنص المنـسق، فـالتعبير 5596.ToString("c") سوف ينتج عنه"$5,596.00"، أما التعبير 5596.4499.ToString("c3") فسوف ينتج عنه "$5,596.450"، والتعبير (134.5).ToString("f3") سينتج عنه "134.500".

٢- ليست كل التنسيقات قابلة للتطبيق على كل أنواع الأعداد، فمثلا، لا يمكن تنسيق عدد بتنسيق سداسي عـشري إلا إذا كان عددا صحيحا.

ولكنّك قد لا تجد التنسيق الذي تريده في الرموز السابقة، فماذا تفعل؟

في هذه الحالة يمكنك تكوين صيغة عامة للتنسيق الذي تريده، باستخدام الرموز التالية:

الرمز	معناه
0	خانة عشرية في هذا الموضع، فإذا كانت غير موجودة، يتم وضع صفر مكانهـا، وأنـت تعرف أن الصفر على أقصى يمين العلامة العشرية لا يؤثّر في قيمة العدد، فالعدد ٠,١ هو نفسه العدد ٠,١٠.
#	رقم يؤثّر في قيمة العدد في هذا الموضع (مثل الجزء الصحيح من العدد).
.	علامة عشرية في هذا الموضع.
,	فاصلة في هذا الموضع.
%	علامة مئوية في هذا الموضع.
E+0 E-0 e+0 e-0	لتنسيق الرقم في صيغة أسية.. ولقد ذكرنا سابقا أن العدد ٢٣٣ × ١٠ ^{١٢} يمثّله الكمبيـوتر كالتالي: 233E+12.
\n	سطر جديد، بمعنى أن الجزء التالي من النص سيعرض في سطر جديد.
" "	لعرض نص بين علامتي تنصيص.

مثال:

```
MsgBox (123.4890.ToString("#.00")) ;
'ستعرض الرسالة الرقم مقربا إلى أقرب خانتين عشريتين: ١٢٣,٤٩
```

ما زالت لدينا نقطة هامة، هي الرموز المستخدمة لتنسيق التواريخ، وسيوضحها لك الأمثلة التالية، حيث وضعنا الناتج كتعليق بجوار كل جملة:

```
lblDATE.Text (birthDate.ToString("d"));
'1/1/2004
lblDATE.Text (birthDate.ToString("D"));
'Saturday, January 01, 2004
lblDATE.Text (birthDate.ToString("f"));
'Saturday, January 01, 2004 12:00 AM
lblDATE.Text (birthDate.ToString("s"));
```

```
'2004-01-01T00:00:00
lblDATE.Text (birthDate.ToString("U"));
'Saturday, January 01, 2004 12:00:00 AM
```

ملحوظة١:

لاستعادة الرقم الأصلي من النص المنسق، يمكنك استخدام الوسيلة Parse، التي تمتلكها كـلّ أنـواع البيانـات الرقمية.

ملحوظة٢:

هناك مجموعة من دوال التنسيق الخاصة بـ #C ما زال بإمكانك استخدامها حتّى الآن.. هذه الدوالّ هي: تنسيق **Format** ـ تنسيق العملة **FormatCurrency** ـ تنسيق التاريخ والوقت **FormatDateTime** ـ تنسيق الأرقام **FormatNumber** ـ تنسيق مئوي **FormatPercent**.. وسأترك لك مهمة استكشاف هـذه الدوالّ

التحويل بين أنواع المتغيرات Converting Variable Types:

في الجدول التالي، ستجد الدوال التي تسمح لك بالتحويل من نوع متغير لآخر:

الدالة	تحول إلى
CBool	منطقي Boolean
CByte	وحدة ذاكرة Byte
CChar	حرف Unicode character
CDate	تاريخ Date
CDbl	عدد مزدوج Double
CDec	عدد عشري Decimal
CInt	عدد صحيح (Int32) Integer
CLng	عدد طويل (Int64) Long
CObj	كائن Object
CShort	عدد قصير (Int16) Short
CSng	عدد مفرد Single
CStr	نص String

مثال:

لتحويل عدد صحيح إلى عدد مزدوج، جرب ما يلي:

```
Integer A;
Double B;
B = CDbl(A);
```

ولكن فيم نحتاج ذلك؟

افترض أنّك عرفت عددين صحيحين كالتالي:

```
Integer A, Integer B ;
A = 23
B = 7
```

لو حاولت قسمة العددين ٧/٢٣، فسيكون الناتج عددا مزدوجا.. هنا سينتج خطأ عن جملةٍ كالتالية:

```
A = A / B
```

ولكن استخدام دالة التحويل، يمكن أن يُخلصك من هذا الخطإ:

```
A = CInt(A / B)
```

وإن كان الثمن هو أنّك ستفقد الدقّة العشرية للناتج، ولو لم تكن تريد ذلك، فعليك أن تعرف متغيرا مزدوجا تستقبل فيه الناتج:

```
Double C;
C = A / B
```

كما أن بإمكانك استخدام الدالة العامة "تحويل النوع" CType، للتحويل إلى أي نوع تحدده لها.. إليك هذا المثال:

```
String A = "34.56";
Double  B ;
B = CType(A, Double) / 1.14;
```

مجال المتغير Variable Scope:

مجال المتغير هو مقطع الكود الذي يمكن خلاله رؤيةُ المتغير واستخدامه.

وهذا المجال ببساطة، هو أصغر مقطع يوجد فيه المتغير ويُسمى المتغير المعرف على مستوى الإجراء أو الدالة متغيرا موضعيا Local Variable، بينما يسمى المتغير الذي نعرفه داخل أحد المقاطع متغيرا مقطعيا Block Variable.. أما المتغيرات المعرفة على مستوى الخلية أو القالب، فتسمى متغيرات مستوى القالب Module-level Variables.

لمعرفة حجم كل متغير من حيث أصغر قيمة أو أكبر قيمة جرب التالى :

```
Response.Write (int.MinValue );
Response.Write (int.MaxValue );
Response.Write (long.MinValue);
Response.Write (long.MaxValue);
Response.Write (float.MinValue);
```

ملحوظة:

لو أردت تجربة أي كود موجود في أمثلة هذا الفصل، فأنشئ صفحة جديدة، وضع على النموذج زرا، وانسخ الكود إلى حدث ضغط هذا الزر.

الشروط الواجب توافرها عند تعريف المتغيرات:

- ألا يكون كلمة من كلمات اللغة الأساسية (تلك التي تراها باللون الأزرق).

- ألا يزيد عن ٢٥٥ حرفًا، وهو رقم كبير بالفعل بما يكفي.
- أن يتكون من كلمة واحدة لا تتخللها المسافات.. ويمكن استخدام الشرطة المنخفضة "_" للفصل بين مقاطع الكلمة بدلا من المسافات.
- لا يبدأ بأرقام، وإن كان من الممكن أن تتوسطه أرقام، أو ينتهي بها.
- لا يحتوي على أيٍ من: علامات التنصيص أو الأقواس أو النقطة "."، ولا علامات العمليات الحسابية أو علامات المقارنة الحسابية أو المنطقية، فكل هذه العلامات محجوزة لوظائف أخرى.
- غير مسموح بتكرار اسم المتغير داخل نفس النطاق، فلا يمكن تعريف متغيرين متماثلين في الاسم داخل نفس الإجراء، وإن كان من الممكن تكرار نفس اسم المتغير لكن في إجراءات مختلفة.

والمتغيرات في لغة #C لا تتجاهل حالة الأحرف Case-insensitive، فالأسماء myAge و myage و MYAGE، كلّها ليست متكافئة، ولاتشير لنفس المتغير.. معنى هذا أنّك تستطيع استخدام هذه الكلمات لتعريف ثلاثة متغيرات مختلفة.

المصفوفاتArrays

البرمجة الحقيقية تبدأ من هذه النقطة، فلقد صُنع الكمبيوتر أساسا، ليقوم بالعمليات الرتيبة المتكررة لآلاف أو ملايـين المرات، بسرعةٍ وبدون ملل.

افترض مثلا أنّك تريد حساب متوسط العمر لخمسين طالبا.. أول ما ستفكّر فيه، هـو أن تعـرف خمـسين متغيـرا وتجمعها معا وتقسم الناتج على ٥٠.. إن مثل هذه الطريقة ستستهلك منك شهرا على الأقلّ لكتابتها، وهي كفيلة بجعلك تكره البرمجة على بكرة أبيها!

مع أنّك تستطيع أن تكتب هذا البرنامج في خمس سطور لا غير.. تخيل!

والفكرة كلّها تعتمد على تخزين أعمار الطلبة في "تركيب ما"، يمكن للكمبيوتر أن يتعامل معه بطريقـة آليـة رتيبـة متكررة، لينفّذ عليه العمليات التي نريدها.

هذا التركيب هو المصفوفة Array، التي تتكون من مجموعة من الخانات، كل خانة منها تحتفظ بقيمة معينة، بحيـث يمكن الوصول لهذه القيمة عن طريق رقم خانتها Index.

ألا يبدو لك هذا الكلام مألوفا؟

إن النص هو عبارة عن مصفوفة من الحروف، ولقد استخدمنا بالفعل الدالة "حروف" Chars للحصول على أي حرف من أي موضع في مصفوفة النص:

ولكن كيف نعرف المصفوفة؟

تماما كما تعرف المتغير العادي، ولكنّنا سنُتبع اسمه بقوسين يحتويان على عدد خانات المصفوفة:

```
String[  ] drinkList = new String[4];
```

طريقة اخرى

```
object[ ] objPersonalInfo;
objPersonalInfo = new object[10];
objPersonalInfo[0] = "James Foxall";
objPersonalInfo[1] = "Papillion";
objPersonalInfo[2] = "Nebraska";
objPersonalInfo[3] = 32;
```

كل خانات المصفوفة لها نفس النوع، ولكن لو عرفت مصفوفة من النوع "كائن" Object، فيمكنك أن تضع في كل خانة قيمة مختلفة في النوع عن الأخرى.

المصفوفة متعددة الأبعاد Multidimensional Arrays:

كل ما تعاملنا مع من المصفوفات حتّى الآن، هو المصفوفات أحادية البعد.. ولكنّك كثيرا ما ستحتاج لاستخدام مصفوفات متعددة البعد، ثنائية وثلاثية وأكثر

```
int[ , ] intMeasurements;
intMeasurements = new int[3,2];
intMeasurements[0,0] = FirstPersonsHeight;
intMeasurements[0,1] = FirstPersonsWeight;
```

مثال :

```
<html>
<head>
<script runat="server" language="C#">
void Page_Load() {
//تعريف المصفوفة
String[] drinkList = new String[4];
//نضع بعض العناصر داخل المصفوفة و لا حظ انها تبدأ من صفر و تنتهى عند ٣ و لكن مجموع العناصر ٤
كما عرفنا المصفوفة
drinkList[0] = "Water";
drinkList[1] = "Juice";
drinkList[2] = "Soda";
drinkList[3] = "Milk";
//للوصول لعنصر فى المصفوفة من خلال مكانة فى فهرس المصمفوفة
lblArrayList.Text = drinkList[1];
}
</script>
</head>
<body>
<form runat="server">
<asp:Label id="lblArrayList" runat="server"/>
</form>
</body>
</html>
```

الدوال او الوظائف Functions

الدوال او الوظائف هى مجموعة من الاكواد البرمجية التى قد تتكرر كثيرا فى البرنامج لذلك نقوم بعمل دالة نعطيها المتغيرات فتقوم بأجراء العمليات عليهم ثم تخرج لنا معلومات جديدة

ونعرف الدالة كما نعرف المتغير العادى ثم نفتح قوسين ( ) ثم نكتب الكود داخل

{الكود}

```
string getName() {
```

```
return "Zak Ruvalcaba";
}
```

مثال :

```
<html>
<head>
<script runat="server" language="C#">
//لاحظ بداية الدالة
string getName() {
return "Zak Ruvalcaba";
}
//الان سوف نستخدمها فى حدث تحميل الصفحة
void Page_Load() {
lblMessage.Text = getName();
}
</script>
</head>
<body>
<form runat="server">
<asp:Label id="lblMessage" runat="server" />
</form>
</body>
</html>
```

لكن لاحظ عند تعريف الدالة يجب ان تحدد نوع البيانات العائدة من الدالة و لنفترض ان لديك دالة من نوع int و اسمها addUp لكى تعمل يجب ان نعطيها متغيران من النوع int هما x , y ثم بعد ذلك تقوم بعملية الجمع x+y لاحظ ان الناتج هو int لذلك لكى نقوم بكتابتة الناتج فى صفحة الويب يجب تحويل الناتج الى متغير نصى من خلال الوظيفة ToString

مثال

```
int addUp(int x, int y) {
return x + y;
}
//الاستدعاء الدالة فى حدث تحميل الصفحة
void Page_Load() {
lblMessage.Text = Convert.ToString(addUp(5, 5));
}
```

لاحظ طريقة استدعاء الدالة نكتب اسم الدالة addUP ثم نفتح قوس و تكتب بة قيمة المتغيرات التى سبق تعريفها قبل كذلك عند كتابة الدالة و لاتنسي ان تفصل بين قيمة كل متغير بفاصلة .

المعاملات Operators

اكبر من	>
اكبر من او يساوى	>=
اقل من	<
اقل من او يساوى	<=

لا تساوى	!=
يساوى	==
اسناد قيمة لمتغير	=
اوor	||
مع and	&&
جمع	+
تخليق كائن جديد	New
ضرب	*
قسمة	/
طرح	-

مثال :

```
if (user == "mic" && itemsBought != 0) {
lblMessage.Text = "Hello mic! Do you want to proceed to " +
"checkout?";
}
```

العمليات المنطقيةConditional Logic

جملة الشرط If…:

تستطيع أن تختبر حدوث شرط معين، فإذا كان صحيحا يتم تنفيذ مقطع الشرط، وإن كان خاطئا يقفز التنفيذ إلى جملـة نهاية الشرط.

ولكن ما هو الشرط؟

الشرط هو أي متغير منطقي أو أي تعبير يعطي نتيجة منطقية (True أو False)

مثال :

```
if (txtUsername.Text == "Zak") {
Response.Redirect("ZaksPage.aspx");
} else {
Response.Redirect("errorPage.aspx");
}
```

جملة اختيار الحالة switch case:

هذه حالة خاصة من جملة الشرط، لتسهيل كتابة الكود عندما تكون كل المقارنات بين متغير واحد، ومجموعة من القيم من نفس نوع المتغير.. ويوضع اسم المتغير بعد جملة case **switch**، بينما توضع القيم المختلفة واحدة بواحدة بعـد كلمة case، متبوعة بالمقطع الذي سيتم تنفيذه في حالة تحقّق التساوي:

```
switch (متغير)
{
    Case قيمة١
        مقطع١ (يتم تنفيذه حينما يساوي التعبير القيمة١)
    break;
    Case قيمة٢
        مقطع٢ (يتم تنفيذه حينما يساوي التعبير القيمة٢).
        ......................
        ......................
    break;

    default:
        مقطع_أخير (يتم تنفيذه حينما لا يساوي المتغير أي قيمة مما سبق)
}
```

```
string strName = txtUsername.Text;
switch (strName) {
case "mic":
Response.Redirect("micPage.aspx");
break;
case "Mak":
Response.Redirect("MarkPage.aspx");
break;
case "Fred":
Response.Redirect("FredsPage.aspx");
break;
default:
Response.Redirect("errorPage.aspx");
break;
}
```

تراكيب التكرار Loop Structures

جملة التكرار "من إلى" For:

هذا هو التركيب التكراري الذي يتطلّب أن تعرف عدد المرات التي سيتم تنفيذه فيها، حيث يعتمد على وجـود عـداد للتكرار Counter، تحدد له قيمة البداية، وقيمة النهاية، والخطوة Step التي سيقفزها (القيمة التي سيزيد بهـا) كلّمـا وصل التنفيذ للشرط

```
int i;
for (i = 1; i <= ddlProducts.Items.Count; i++) {
```

```
lblMessage.Text = Convert.ToString(i);
}
```

لاحظ انة لكى تخرج من التركيب التكرارى يمكنك استخدام الامر break

```
for (i = 0; i <= 10; i++) {
if (i == 5) {
Response.Write("Oh no! Not the number 5!!");
break;
```

تركيب التكرار "بينما" While :

While(شرط)

{مقطع

}

مثال :

```
int counter = 0;
while (counter <= 10) {
```

لاحظ طريقة التحويل من int الى string //

```
lblMessage.Text = counter + "";
```

لاحظ لغة #C تستخدم المعامل ++ للزيادة بمقدار ١//

```
counter++;
}
```

تركيب التكرار "نفّذ مرارا" Do… while:

هذا التركيب أيضا ينفّذ مقطعا من الكود، ولكنّه أعم من متكررة …For، لأنّه يعتمد على صحة شرط ما لاستمرار التكرار، بدلا من الاعتماد على قيمة العداد.. إن هذا يُمكّنك من تعقيد هذا الشرط لأي درجة.

```
int counter = 0;
do {
lblMessage.Text = counter + "";
counter++;
} while (counter <= 10);
```

تركيب التكرار foreach:

نوع من انواع for لكنة يدور داخل كل عنصر فى المحتوى مثال البحث عن عنصر معين فى مصفوفة اسمها arrayName

```
foreach (string item in arrayName) {
lblMessage.Text = item;
}
```

الان سنتكلم عن لغة asp.net

كائنات ASP.NET

لا أعتقد أنّك تستطيع كتابة التطبيقات التي تريدها، دون أن تتعرف على الكائنات التالية:

كائن الصفحة Page Object:

يمثّل هذا الكائن الصفحة المعروضة على جهاز العميل.. ولهذا الكائن ـ مثل باقي الكائنات ـ حدثان: حدث التحميـل Load، وحدث الإغلاق Unload.

وأهم خاصية لكائن الصفحة، هي خاصية "تم إرجاعه" IsPostBack، التي تعطيك القيمة True إذا تم إرسال بيانـات الصفحة إلى الخادم.. وتفيد هذه الخاصية في تحديد إذا ما كانت الصفحة تحتاج لوضع القيم الابتدائية فـي أدواتهـا أم لا.. وكما اتفقنا سابقا، من المحبذ الاحتفاظ بمدخلات المستخدم بعد إرسال بياناتها إلى الخادم.

كائن الاستجابة Response Object:

هذا الكائن مشتقّ من الخلية HttpResponse.. وهو يمكّنك من إرسال النتائج من الخادم إلى العميل.

أهم الخصائص:

تخزين Buffer:

لو جعلت قيمة هذه الخاصية True، يقوم الخادم بتخزين النتائج المطلوبة إلى أن تكتمل، ثم يرسلها دفعةً واحـدةً إلـى العميل.. بخلاف ذلك، يتم إرسال النتائج مجزئةً، كلّما اكتمل منها جزء.. والوضع الأخير في الغالب أفضل، لأنّه يقلّل الوقت الذي ينتظره المستخدم.

البيانات المؤقّتة Cookies:

مجموعة Collection تمثّل البيانات التي يخزنها الخادم على جهاز العميل ـ إن كان العميل يتيح هـذه الإمكانيـة.. (وسنناقش ذلك بالتفصيل لاحقا).

الزوال Expires:

تقرأ هذه الخاصية أو تغير المدة الزمنية (بالدقيقة)، التي ستبقى فيها الصفحة مهيأة Cached على جهاز العميـل.. إن الصفحات المهيأة لا يتم تحميلها للعميل في كلّ مرة، مما يوفّر الكثير من الوقت.

الزوال المحدد ExpiresAbsolute:

مماثلة للخاصية السابقة، ولكنّها لا تتعامل بالدقائق، بل بالتاريخ والوقت الذي سيزول بعدها الصفحة المهيأة من علـى جهاز العميل.

هل العميل متّصل IsClientConnected:

تعيد هذه الخاصية True إذا كان العميل ما زال متّصلا بالخادم.. استخدم هذه الخاصية في البرامج التي تأخـذ وقتـا طويلا في التنفيذ، فمن الممكن أن يغلق العميل الصفحة (من فرط الملل!!)، وبهذا يصبح استمرارك في تنفيذ الكود بلا طائل.

الوسائل او الدوال :

محو المحتوى ClearContent:

تزيل كلّ المخرجات الموجودة في مخزن المخرجات Output Buffer Stream.

محو العناوين ClearHeaders:

تمحو كلّ العناوين Headers من مخزن المخرجات.

إغلاق Close:

تغلق الاتصال مع العميل.

إنهاء End:

تنهي تنفيذ المهمّ' المطلوبة حاليا، ويتم إرسال المخرجات الموجودة في المخزن Buffer للعميل.

محو Flush:

ترسل المخرجات الموجودة في المخزن Buffer للعميل.. استخدم هذه الوسيلة لإرسال البيانات للعميل على فترات، إذا استغرق التنفيذ وقتا طويلا.

إعادة التوجيه Redirect:

تقبل هذه الوسيلة عنوان أحد المواقع URL، وتصل المستخدم به.. وفي الغالب، تستخدم هذه الوسيلة لعرض الصفحات في التطبيقات التي تتكون من أكثر من صفحة.

كتابة Write:

استخدم هذه الوسيلة لإضافة نصوص (أو حتّى كائنات) للمخرجات التي سيتم إرسالها للعميل.

كتابة ملفّ WriteFile:

استخدم هذه الوسيلة لإرسال محتويات ملفّ موجود على الخادم إلى العميل.

كائن الاستعلام Request Object:

هذا الكائن مشتقّ من الخلية HttpRequest.. وهو يسمح لك بالتعامل مع الاستعلامات الواردة من العميل إلى الخادم.. وفيما يلي أهم خصائص هذا الكائن:

مسار التطبيق ApplicationPath:

تعيد المسار الذي يعمل عليه التطبيق على الخادم.

طول المحتوي ContentLength:

طول البيانات المرسلة من العميل (بالبايت).

البيانات المؤقّتة Cookies:

تعيد مجموعة Collection تحتوي على البيانات المؤقّتة المرسلة من العميل.

العنوان الخام RawUrl:

تعيد العنوان URL المطلوب حاليا.

مسار الملفّ FilePath:

تعيد المسار الافتراضي Virtual Path للاستعلام الحالي.

المسار الفعلي PhysicalPath:

تعيد مسار الصفحة التي تمثّل العنوان URL المطلوب.

هل الاتصال مؤمن IsSecureConnection:

تعيد هذه الخاصية True إذا كان الاتصال يستخدم وصلات مؤمنة secure sockets، مثل HTTPS.

متغيرات الخادم ServerVariables:

تعيد مجموعة Collection تحتوي على متغيرات الخادم.

حجم البيانات TotalBytes:

تعيد المجموع الكلّي للبيانات الموجودة في الاستعلام (بالبايت).

العنوان Url:

تعيد معلومات عن العنوان المطلوب حاليا.

العنوان السابق UrlReferrer:

تعيد العنوان السابق، الذي ضغط من خلاله المستخدم الوصلة الفائقة التي فتحت العنوان الحالي.

عنوان المضيف UserHostAddress:

تعيد معرف عنوان العميل IP.

اسم المضيف UserHostName:

تعيد الاسم DNS name الخاص بالعميل.

لغة العميل UserLanguages:

تعيد مصفوفة، تحتوي على أسماء اللغات المفضلة لدى العميل.

المتصفّح Browser:

هذه الخاصية عبارة عن كائن يمنحك المعلومات اللازمة عن المتصفّح الذي يعمل على جهاز العميل.. ومن خصائص هذا الكائن:

AOL	تعيد True إذا كان المتصفّح متصفّحا أمريكيا فوريا America Online.

BackgroundSounds	تعيد True إذا كان المتصفّح يسمح بإصدار الأصوات في خلفية التشغيل.
Beta	تعيد True إذا كان المتصفّح إصدارا تجريبيا.
Cookies	تعيد True إذا كان المتصفّح يستطيع تخزين البيانات المؤقّتة على جهـاز العميـل.. ولا علاقة لقيمة هذه الخاصية بتفعيل المستخدم أو إغلاقة لإمكانية تخـزين البيانـات المؤقّتـة Cookies على جهازه.
Crawler	تعيد True إذا كان المتصفّح محرك بحث Crawler Search Engine.
Frames	تعيد True إذا كان جهاز العميل يستطيع التعامل مع إطارات HTML.
JavaApplets	تعيد True إذا كان المتصفّح يستطيع التعامل مع Java Applets.
VBScript	تعيد True إذا كان المتصفّح يستطيع التعامل مع الكود المكتوب بلغة VBScript.
JavaScript	تعيد True إذا كان المتصفّح يستطيع التعامل مع الكود المكتوب بلغة JavaScript.
MajorVersion	رقم الإصدار الرئيسي للمتصفّح.
MinorVersion	رقم الإصدار الفرعي للمتصفّح.
Platform	اسم نظام التشغيل الذي يعمل عليه المتصفّح.
Tables	تعيد True إذا كان المتصفّح يستطيع التعامل مع جداول HTML.

مثال :

```
<%@ PAGE language='c#' debug='true' trace='false' %>
<HTML>
<HEAD>
<TITLE>ASP.NET Browser Capabilities</TITLE>
</HEAD>
<BODY>
<table width="350" border="0" cellspacing="1" cellpadding="3"
  bgcolor="#000000">
 <tr bgcolor="#FFFFFF">
  <td>Supports ActiveX Controls</td>
  <td><% =Request.Browser.ActiveXControls %></td>
 </tr>
 <tr bgcolor="#FFFFFF">
  <td>Is an America Online Browser</td>
  <td><% =Request.Browser.AOL %></td>
 </tr>
 <tr bgcolor="#FFFFFF">
  <td>Supports background sounds</td>
  <td><% =Request.Browser.BackgroundSounds %></td>
 </tr>
 <tr bgcolor="#FFFFFF">
  <td>Is a beta version browser</td>
  <td><% =Request.Browser.Beta %></td>
 </tr>
```

```
<tr bgcolor="#FFFFFF">
 <td>Browser name (user-agent)</td>
 <td><% =Request.Browser.Browser %></td>
</tr>
<tr bgcolor="#FFFFFF">
 <td>Supports Channel Definition Format</td>
 <td><% =Request.Browser.CDF %></td>
</tr>
<tr bgcolor="#FFFFFF">
 <td>Common Language Runtime version</td>
 <td><% =Request.Browser.ClrVersion %></td>
</tr>
<tr bgcolor="#FFFFFF">
 <td>Cookies available</td>
 <td><% =Request.Browser.Cookies %></td>
</tr>
<tr bgcolor="#FFFFFF">
 <td>Is this a Web search engine (&quot;crawler&quot;)?</td>
 <td><% =Request.Browser.Crawler %></td>
</tr>
<tr bgcolor="#FFFFFF">
 <td>Version of JavaScript (ECMAScript) supported</td>
 <td><% =Request.Browser.EcmaScriptVersion %></td>
</tr>
<tr bgcolor="#FFFFFF">
 <td>Supports frames</td>
 <td><% =Request.Browser.Frames %></td>
</tr>
<tr bgcolor="#FFFFFF">
 <td>Supports client-side Java</td>
 <td><% =Request.Browser.JavaApplets %></td>
</tr>
<tr bgcolor="#FFFFFF">
 <td>Supports JavaScript (ECMAScript)</td>
 <td><% =Request.Browser.JavaScript %></td>
</tr>
<tr bgcolor="#FFFFFF">
 <td>Browser version</td>
 <td>
  <% =Request.Browser.MajorVersion + "." +
   Request.Browser.MinorVersion %>
 </td>
</tr>
<tr bgcolor="#FFFFFF">
 <td>Microsoft XML Document Object Model version</td>
 <td><%=Request.Browser.MSDomVersion %></td>
</tr>
<tr bgcolor="#FFFFFF">
```

```
    <td>Operating system platform</td>
    <td><% =Request.Browser.Platform %></td>
  </tr>
  <tr bgcolor="#FFFFFF">
    <td>Supports HTML tables</td>
    <td><% =Request.Browser.Tables %></td>
  </tr>
  <tr bgcolor="#FFFFFF">
    <td>Client browser type</td>
    <td><% =Request.Browser.Type %></td>
  </tr>
  <tr bgcolor="#FFFFFF">
    <td>Browser supports VBScript</td>
    <td><% =Request.Browser.VBScript %></td>
  </tr>
  <tr bgcolor="#FFFFFF">
    <td>Version of client browser</td>
    <td><% =Request.Browser.Version %></td>
  </tr>
  <tr bgcolor="#FFFFFF">
    <td>W3C HTML Document Object Model version</td>
    <td><% =Request.Browser.W3CDomVersion %></td>
  </tr>
  <tr bgcolor="#FFFFFF">
    <td>Running 16-bit Windows?</td>
    <td><% =Request.Browser.Win16 %></td>
  </tr>
  <tr bgcolor="#FFFFFF">
    <td>Running 32-bit Windows?</td>
    <td><% =Request.Browser.Win32 %></td>
  </tr>
</table>
</BODY>
</HTML>
```

عند تشغيل الصفحة تظهر النتائج التالية :

Supports ActiveX Controls	True
Is an America Online Browser	False
Supports background sounds	True
Is a beta version browser	False
Browser name (user-agent)	IE
Supports Channel Definition Format	False
Common Language Runtime version	1.1.4322
Cookies available	True

Is this a Web search engine ("crawler")?	False
Version of JavaScript (ECMAScript) supported	1.2
Supports frames	True
Supports client-side Java	True
Supports JavaScript (ECMAScript)	True
Browser version	6.0
Microsoft XML Document Object Model version	6.0
Operating system platform	WinNT
Supports HTML tables	True
Client browser type	IE6
Browser supports VBScript	True
Version of client browser	6.0
W3C HTML Document Object Model version	1.0
Running 16-bit Windows?	False
Running 32-bit Windows?	True

كائن الخادم Server Object:

يمنحك هذا الكائن الكثير من المعلومات المهمة عن الخادم.. فمثلا، يمنحك خاصية "اسم الجهـاز" MachineName، التي تعيد اسم الجهاز الذي يعمل عليه الخادم.

وسائل كائن الخادم:

GetLastError:

تعيد آخر استثناء Exception حدث أثناء التنفيذ.

ClearError:

تمحو بيانات آخر خطإ قد حدث.

خريطة المسارات MapPath:

تعيد المسار الكامل للملفّ المرسل لها كمعامل.. ولقد رأينا كيفية استخدامها من قبل.

تنفيذ Execute:

تنفّذ كود الصفحة، التي تم إرسال عنوانها كمعامل، دون أن تعرض هذه الصفحة.. ويتم عرض ناتج الصفحة الجديـدة على الصفحة القديمة (كما سنرى لاحقا).

انتقال Transfer:

تلغي تنفيذ الصفحة الحالية وتنتقل إلى صفحة أخرى، دون أن تتم العودة للصفحة السابقة.

تشفير العنوان UrlEncode:

لو جربت المثال التالي:

```
Server.UrlEncode("who am I?");
```

فستحصل على النص:

who+am+I%3f

فك تشفير العنوان UrlDecode:

تفك تشفير النص الذي تحصل عليه من الوسيلة السابقة.

تشفير HtmlEncode:

تعمل على تشفير كود HTML المرسل لها كمعامل.

فك التشفير HtmlDecode:

تعمل على فك تشفير النص الناتج من الوسيلة السابقة.

تشفير مسار العنوان UrlPathEncode:

تشفّر الجزء الخاص بالمسار في عنوان الموقع.

استخدام البيانات المؤقّتة Cookies:

قلنا إن بإمكانك التعامل مع مجموعة البيانات المؤقّتة Cookies collection الموجودة حاليا على جهاز المستخدم، عبر خاصية Cookies، التي يمنحها لك كائنا الاستعلام Request والاستجابة Response.

وكأي مجموعة Collection، يمكنك إضافة المزيد من البيانات المؤقّتة لجهاز المستخدم.. ابدأ أولا بتعريف هذه البيانات كالتالي:

```
HttpCookie hc = new HttpCookie ("User");
```

حيث User هو المفتاح الذي تشير به إلى هذه البيانات المؤقّتة (يمكن أن يكون أي نص آخر).

الآن يمكنك استخدام خصائص كائن البيانات المؤقّتة التالية:

Value	البيانات المؤقّتة التي ستخزنها على جهاز المستخدم.
Values	مجموعة Collection تستخدم في حالة وضع بيانات عديدة في نفس الكائن.. في هذه الحالة يكون لكلّ قيمة مفتاح Key يتم التعامل معها من خلاله.
Expires	التاريخ الذي سيقوم جهاز العميل بمحو هذه البيانات عنده.. ولو تحدد هذا التاريخ، فسيتم محو هذه البيانات فور إغلاق التطبيق.
Path	المسار الذي سيتم تخزين هذه البيانات عليه.

تعال نضع قيمة في الكائن الذي عرفناه:

```
hc.Value = txtCookie1.Text;
```

وكان من الممكن اختصار الخطوة السابقة، بتعريف كائن البيانات المؤقّتة منذ البداية كالتالي:

```
HttpCookie hc = new HttpCookie ("User", txtCookie1.Text);
```

الآن تعال نضيف هذا الكائن لمجموعة البيانات المؤقّتة الموجودة على جهاز العميل:

```
Response.Cookies.Add(hc);
```

ملحوظة:

رغم أن باستطاعتنا إضافة البيانات المؤقّتة لكائن الاستعلام Request، إلا إن ذلك لن يؤثّر على جهـاز المـستخدم.. كائن الاستجابة Response فقط هو الذي يؤثّر على جهاز المستخدم.. إذن خذها كقاعدة: لإضافة البيانات المؤقّتـة، استخدم كائن الاستجابة، ولقراءتها استخدم كائن الاستعلام.

ولحذف بعض البيانات المؤقّتة، استخدم الوسيلة Remove، كالتالي:

```
Request.Cookies.Remove("User");
```

التعامل مع أكثر من نموذج (صفحة):

أبسط طريقة لعرض صفحة من خلال أخرى، هي استخدام أداة الوصلة الفائقة Hyperlink Control، مع وضع اسم النموذج الذي تريد عرضه في خاصية NavigateUrl لهذه الأداة.

ورغم سهولة هذه الطريقة، إلا إنّها لن تنفعك إذا أردت تمرير بعض المعاملات للصفحة الجديدة بطريقة مباشرة.. في هذه الحالة يمكنك استخدام الطرق التي شرحناها في مقطع "إبقاء الحالة"، مثل تعريف بعض المتغيـرات عبـر كـائن الجلسة Session Object.

أو بإمكانك استخدام إحدى الوسيلتين Response.Redirect أو Server.Execute .

تعال نرى كيف نستخدم الوسيلة Redirect.. إن هذه الوسيلة تقبل كمعامل، عنوان الموقع (الـصفحة) التـي تريـد تحميلها.. هنا يمكن أن تضيف المعاملات للنص الذي يمثّل العنوان كالتالي:

- أضف علامة استفهام إنجليزية ? بعد اسم الموقع لتفصله عن المعاملات.
- أضف أزواج المعاملات (الاسم والقيمة) يفصل بينهما علامة =.
- افصل بين كلّ زوجين متتاليين من المعاملات بعلامة &.
- عليك بعد كلّ هذا تشفير النص الناتج باستخدام الوسيلة Server.UrlEncode.

انظر للمثال التالي:

```
String params;
params = "Var1=" + Server.UrlEncode(TextBox1.Text) + " & "
params += "Var2=" + Server.UrlEncode(TextBox2.Text) + " & "
params += "Var3=" + Server.UrlEncode(TextBox3.Text)
Response.Redirect("WebForm3.aspx?" & params);
```

ولكن كيف سنقرأ هذه المعاملات؟

يمكنك في هذه الحالة استخدام خاصية QueryString في حدث تحميل النموذج كالتالي:

استخدم المعاملات في أي شيءٍ تريده

```
Response.Write(Request.QueryString("Var1"));
Response.Write(Request.QueryString("Var2"));
Response.Write(Request.QueryString("Var3"));
```

أما الوسيلة Execute، فهي مختلفة إلى حد بعيد، ذلك أنّها لا تعرض الصفحة الجديدة، ولكنّها تنفّذ ما بها مـن كـود، وتضيف الناتج للصفحة الأولى!

لهذا فإن أسلم طريقة لاستخدام هذه الوسيلة، هي قراءة الناتج منها في كائن من النوع StringWriter، ووضع الناتج في لافتة على النموذج الحالي.

انظر للمثال التالي:

```
Dim pageOut As New System.IO.StringWriter()

Server.Execute("ShowBasket.aspx", pageOut)
Response.Write("<H2>تحتوي سلّتك على هذه العناصر:</H2>")
Response.Write("<BR>")
Response.Write(pageOut.ToString)
```

ولكي تتعامل مع أدوات النموذج الأول من النموذج الثاني، استخدم الكود التالي:

```
Private Sub Page_Load(ByVal sender As System.Object, _
ByVal e As System.EventArgs) Handles MyBase.Load
    Dim key, value As String
    Response.Write(Request.QueryString)
    For Each key In Request.Form.Keys
        Response.Write(key + " = " + Request.Form.Item(key))
        Response.Write("<BR>")
    Next
End Sub
```

الخلايا Classes

تقع الخلايا Classes في موضع القلب من C#.NET، وإطار العمل NET Framework. نفسه ما هو إلا مجموعة هائلة من الخلايا التي يمكنك استدعاؤها واستخدامها في لغة البرمجة التي تعمل بها.

ليس هذا فحسب، فإن صفحة الويب ما هو إلا خلية، تحتوي على الأدوات والكود الذي يتعامل معها.

وكما قلنا من قبل، إن عملية تعريف متغير من أي نوع، ما هي إلا عملية إنشاء نسخة من خلية معينة، بحيث تستطيع استخدام كل خصائص ووسائل هذه الخلية عبر هذا المتغير.

ولكن... ما رأيك أن نتركنا من كل هذا الكلام، لنتعرف على الخلايا بالتفصيل؟

إن الخلية هي برنامج لا يمكن تنفيذه بمفرده، إذ يجب أن يتم استخدامه بواسطة تطبيق آخر أو جزء آخر من نفس التطبيق الذي يحتوي على الخلية.

ستسألني:

- أتعني أنّنا نستدعي الخلية كما نستدعي الدوال؟
- إطلاقا.. إن الخلية تحتوي أساسا على العديد من الإجراءات والدوال والخصائص الخاصة بها.
- إذن كيف نستخدم الخلية؟
- أنّنا نعرف متغيرا من نفس نوع الخلية.
- بالضبط.. وهذا يعني ببساطة، أن الخلية هي إحدى أنواع المتغيرات التي يعرفها المستخدم User Defined Types، وإن كانت هي أشملها وأقواها وأهمها على الإطلاق، كما سنرى بعد قليل.

تنظيم الخلايا داخل فضاءات الأسماء Namespaces:

نظرا للكم الهائل من الخلايا Classes التي يحتوي عليها إطار العمل، فقد تم تقسيم هذه الأنواع إلى مجموعات علـى حسب وظيفتها، كلّ مجموعة منها تندرج تحت فضاء اسم Namespace خاص بها.. ولقد رأينا كيف تندرج خلايـا التعامل مع الملفّات تحت فضاء الاسم IO الذي يندرج بدوره تحت فضاء الاسم System، ورأينا كيف نستورد هـذا الفضاء في بداية الملفّ لاختصار كتابة الكود، وذلك بالجملة التالية:

```
<%@ Import Namespace="System.Io" %>
<%@ Import Namespace="System.Data.OleDb" %>
```

مكتبة الخلايا Class Library:

وهي مكون أساسي لبناء الكود، ليس لها واجهة مرئية، ولكنّها تؤّدى وظائف معينة لبرنامجك.
وباختصار، فإن الخلية هي مجموعةٌ من الدوالّ والإجراءات التي يمكنُ إعادة استخدامها في أي مشروع، بحيثُ يوفّر المبرمجُ على نفسِه مشقّةَ إعادة كتابتها مرة أخرى.

مكتبة أدوات تحكم الإنترنت Web Control library:

كما أنك تستطيع أن تبني أدوات الويندوز الخاصة بك لتستخدمها على نماذج ويندوز، تستطيع أيـضًا أن تبنـي أدوات الإنترنت الخاصة لتستخدمها على نماذج وصفحات الإنترنت.

البرمجة بالكائنات

مفهوم الكائن : الكائن object هو أجزاء مترابطة تضم بيانات و كود يعمل فى هذة البيانات أما الفئات Class فيمكن اعتبارها الخلايا حيث يمكن من خلالها تخليق كائنات على كل نعتبر دائما بأن الخلايـا هـى اوصـاف للكائنـات و الكائنات التى تخلق من هذة الطبقات تعتبر مثيلات Instances من الخلايا.
علية يمكن القول ان Class تصف الهيكل الداخلى للكائن object التى يحتويها .
من اهم الموضوعات فى البرمجة بالكائنات :

تعدد الأشكال Polymorphism

معناه:

من السمات الرائعة في البرمجة بالكائنات Objects، الإمكانية المسماة بـ "تعدد الأشكال" Polymorphism.. وهذه الإمكانية تعني أنّه بينما يكون لديك العديد من الكائنات المختلفة، ستكتشف أن معظم خصائصها Properties ووظائفها Methods وأحداثها Events مشتركة في الأسماء ومتشابهة في الوظيفة مع بعض الاختلافات الطفيفة.
فمثلا ستجد أن النموذج ومربع النص والقائمة المركّبة، لكل منها خاصية "نص" Text، يمكنك أن تكتب الـنص فيهـا الذي تريدها أن تعرضه.. ولكنك ستلاحظ أن النموذج يعرض النص في شريط بأعلاه، بينما يعرضه مربع النص في داخله، بينما تعرضه القائمة المركبة في مربع النص الخاص بها.. إذن فهناك اختلافات في وظيفة هذه الخاصية فـي كل من هذه الكائنات.. الجميل أنك معزول عن هذه الاختلافات، وليس مطلوبا منك أي شيءٍ للتعامل معها.

هذه ميزة كبيرة، فهي توفّر عليك وقتا طويلا للتعلم، فبمجرد أن تعرف وظيفة خاصية Text في مربع النص مثلا، ستكتشف أنّك قد تعلمت وظيفتها في أي أداةٍ أخرى لها نفس الخاصية، مثل "اللافتة" Label وزر الاختيار CheckBox وزر التحويل Radio Button... إلخ.

يفيد هذا أيضا في سهولة حفظ أسماء الخصائص ووظائفها، فلو كان لنفس الوظيفة اسم مختلف في كل أداة لكانت كارثة!

قليل من الخصائص المشتركة A Few Common Properties:

خاصية "الاسم" Name:

وهذه الخاصية تظهر في أعلى نافذة الخصائص دون مراعاة للترتيب الأبجدي، وذلك لأهميتها.

وهي عبارة عن الاسم البرمجي للأداة، وهو الذي تكتبه في الكود.. مثال:

لو أسميت مربع نص Text1، يمكنك أن تغير النص المكتوب فيه برمجيا بجملة كالتالية:

Text1.Text = "أي شيء";

أي أن هذه الخاصية هي مفتاحك لاستخدام باقي خصائص ووسائل الأداة.

ملاحظة هامة:

نظرًا لأن اسم الأداة سيظهر كثيرًا في البرنامج، يجب عليك اختيار اسم يكون موحيا، حتى يمكنك فهم البرنامج عند قراءته فيما بعد.. والطريقة المتفق عليها في التسمية هي تكوين الاسم من مقطعين:

١- المقطع الأول: بادئة تدل على نوع الكائن، عبارة عن حرفين أو ثلاثة لاختصار اسمه.. وسنعطيك بعض الاقتراحات في الجدول التالي:

Bt	Button
Chk	CheckBox
Cmb	ComboBox
Frm	Form
Lb	Label
Lst	ListBox
Mnu	Menu
Rdo	RadioButton
Rt	RichTextBox
Txt	TextBox

وهكذا...

٢- المقطع الثاني: كلمة تدل على وظيفة الأداة، حتى يمكن فهمها بمجرد قراءة الاسم.. أمثلة:

- لو لديك مربع نص يكتب فيه المستخدم اسمه، فسمه TxtUserName أو TxtName لو أردت الاختصار.
- ولو لديك لافتة تعرض مجموع عددين، فسمها LbSum.
- ولو لديك قائمة مركبة تعرض أسماء الدول العربية، فسمها CmbArabCountries.

- ولو لديك زر مكتوب عليه "موافق"، فسمه BtOk.
وهكذا.... .

خاصية "من اليمين لليسار" RightToLeft:

وهي من أهم الخصائص لنا كمبرمجين عرب، حيثُ إنّها تسمحُ لك باستخدام الأداة في التطبيقـاتِ العربيـة، فتكـون محاذاتها من اليمين لليسار، ويمكنك كتابةُ النصوص العربية بها بدون مشاكل.

خاصية "الخط" Font Object:

وهي تحدد المواصفات التي سيعرض بها الخط في الأداة، في وقتي التصميم والتنفيذ، كاسم الخطّ وحجمه وهـل هـو سميك أو مائل... إلخ.

خاصية "لون الخلفية" BackColor:

تحددُ لون خلفية الأداة.. وقد رأينا في بدايةِ هذا الفصل كيفية اختيار الألوان من نافذة الخصائص.. ويمكن كذلك تغيير هذه الخاصية من الكود.. و

خاصية "لون الخطّ" ForeColor:

تحدد هذه الخاصية لون النص المكتوب في الأداة.. ويمكن تغييرها بطريقة مشابهة لتلك التي يتم بها تغيير لون خلفية الأداة، سواء من نافذة الخصائص، أو من الكود.

خاصية "فعالة" Enabled:

إن الأدوات فعالة في الوضع التلقائي، بمعنى أنّها تستجيب للمستخدم عن طريق الفأرة أو لوحة المفاتيح.. ولكن هنـاك حالات تحتاج فيها لمنع المستخدم من استخدام أداة معينة، حتى لا يتسبب في خطإٍ ما.. فمثلا، لو لديك نافذة تطلب من المستخدم كتابة اسمه وكلمة المرور الخاصة به، وبها زران: "موافق" و"إلغاء"، فمن المنطقي أن تمنع المـستخدم مـن ضغط زر "موافق" إذا لم يكتب الاسم أو كلمة المرور.

هنا يمكنك جعل الزر غير فعال، بكتابة أمر على شاكلة:

Button1.**Enabled** = **false;**

وبهذا يظهر الزر في لون رمادي، ويمتنع عن الاستجابة للمستخدم.

خاصية "الحجم" Size:

لقراءة أو تغيير حجم الأداة.. وهي عبارة عن كائن يحتوي على خاصيتين: "العرض" Width و"الارتفاع" Height.

الخاصية الإضافية Tag:

وهي مخصصة لتخزين أي نوع من المعلومات تريد وضعه داخل الأداة.

خاصية "نص" Text:

تقرأ أو تغير النص الذي يظهر على الأداة.

خاصية "التوقف بضغط زر Tab" TabStop:

أداة واحدة فقط يمكنها أن تكون نشيطة Active في لحظة معينة، بمعنى أنها تستقبل ما يتم ضغطه من الأزرار من لوحة المفاتيح أو الفأرة.. وللانتقال من أداة لأخرى يمكنك أن تضغط الأداة بالفأرة لتنشيطها.. ولكن كثيرًا من المستخدمين يفضلون استخدام لوحة المفاتيح، لأنّها تكونُ أسرع خاصةً عند إدخال كم كبير من البيانات.. لهذا فإن بإمكانهم الانتقال بين الأدوات باستخدام لوحة المفاتيح، وذلك بضغط زر Tab (للتحرك للأداة التالية) أو زري Shift+Tab (للتحرك للأداة السابقة).

هنا يمكنك أن تسمح باستثناء الأداة من هذا التصرف، بجعل قيمة خاصية TabStop خطأ False.. في هذه الحالة لن يستطيع المستخدم الوصولَ إلى الأداة إلا باستخدام الفأرة.

خاصية "ترتيب الانتقال" TabIndex:

وهي قيمة رقمية تستخدم لتوضيح موقع الأداة في "ترتيب الوصول بضغط زر Tab Order "Tab، وهو يبدأ من الصفر.

إن الأداة ذات الرقم الأصغر، هي تلك التي ستستقبل العلامة الضوئية عند بدء ظهور النموذج.

ولو ضغطت زر Tab مرة واحدة، فستنتقل العلامة الضوئية للأداة التي تمتلك أقل رقم يكبر عن الرقم الحالي.

خاصية "مرئية" Visible:

يمكن استخدام هذه الخاصية لعرض أو إخفاء الأداة، بوضع قيمة هذه الخاصية بصواب True أو خطأ False على التناظر.

قليل من الوسائل المشتركة A Few Common Methods:

واضح من اسمها أنها "وسيلة" لأداء وظيفة معينة.. والوسائل هو الاسم الذي يندرج تحته:

-الدالة Function، وهي مثل الإجراء، وإن كنت تستقبل منها قيمة عائدة Return Value.

ولا تحتوي أدوات الويندوز Windows Controls على الكثير من الوسائل، بينما الكائنات Objects تمنح وسائل أكثر ـ كما سنرى فيما بعد.

إن كل شيءٍ ـ حتى المتغيرات بل والأرقام ـ هي كائنات، ولكل منها مجموعة خاصة من الوسائل، يمكن استخدامها بكتابة نقطة "." بعد اسم الكائن، ثم كتابة اسم الوسيلة.. فمثلا يمدنا المتغير النصي String variable بهذه الوسائل:

- "طول النص" Length:

وهي دالة لحساب طول النص.

- "حول للحروف الكبيرة" ToUpper:

وهي دالة تكّون نصًا جديدا كل حروفه كبيرة Capital.

- "حول للحروف الصغيرة" ToLower:

وهي دالة تكّون نصًا جديدا كل حروفه صغيرة Small.

ولترى كيف تعمل هذه الدوال، أنشئ صفحة جديدًا، واكتب الكود التالي في حدث تحميل الصفحة:

```
void Page_Load()

{ Response.Write("Visual C#".Length);

Response.Write("Visual C#". ToUpper);

Response.Write("Visual C#". ToLower);



}
```

اضغط F5 لتشغيل البرنامج، واضغط الزر.. سترى ما يلي

٩

VISUAL C#"

visual c#"

قليل من الأحداث المشتركة A Few Common Events:

إن أوامر برنامجك، تُكتب أساسا في مجيبات الأحداث Event Handlers، حتى يمكـن للبرنـامج أن يتفاعـل مـع المستخدم.

ورغم أن لكل أداةٍ عددا هائلا من الأحداث، فإنك نادرا ما تبرمج أكثر من حدث واحد منها، بل إن كثيرا من الأدوات لا تحتاج لكتابة أي أوامر لها.

وهذه أكثر الأحداث استخدامًا:

حدث "الضغط" Click:

حدث النقر المزدوج DoubleClick:

حدث "الدخول" Enter:

يحدث عندما تستقبل الأداة العلامة الضوئية وتصبح نشيطة.

حدث "الخروج" Leave:

يحدث عندما تغادر العلامة الضوئية الأداة، وتصبح غير نشيطة.

وغالبا ما تكتب هنا الكود الذي يتحقق من صحة محتويات الأداة قبل مغادرتها.

حدث "دخول الفأرة" MouseEnter:

يحدث عندما يعبر مؤشر الفأرة المساحة التي تشغلها الأداة، وهو يطلق مرة واحدة فور دخول مؤشر الفأرة، ولا يطلق مرة أخرى حتى يغادرها ثم يدخلها من جديد.
ولكن لو أردت مراقبة حركة الفأرة فوق الأداة، فعليك باستخدام الحدث "حركة الفأرة" MouseMove.

حدث "مغادرة الفأرة" MouseLeave:

ويحدث عندما يخرج مؤشر الفأرة عن حدود الأداة.

الحدث "تغير..." XXXChanged:

حيث البادئة XXX هي مجرد رمـز عـام، يمكـن اسـتبداله باسـم خاصـية مـا، مثـل TextChanged و BackColorChangedو FontChanged... إلخ.
جلي أن هذه الأحداث تحدث عندما تتغير قيمة خاصية ما سواء غيرها المستخدم، أم غيرها المبرمج من الكود.

ملف الإعدادات web.config

هو أهم ملف في ملفات أي مشروع ASP.NET لأنه يحتوي على جميع إعدادات المشروع مثل تحديد اللغة وجملة الربط بقواعد البيانات والمجلدات التي تريد حمايتها .... الخ وبهذا يمكنك التعديل في خصائص المشروع ككل من خلال التعديل في هذا الملف.
وهذا الملف هو عبارة عن ملف نصي عادي على شكل XML يمكن التعديل عليه بإستخدام أي محرر نصوص مثل notepad .

يمكن أن يحتوي كل مجلد من مجلدات المشروع على ملف web.config ولكن ليس ضروريا لأن ملف web.configيؤثر على المجلدات الفرعية للمجلد الذي يوجد فيه . فمثلا لو كان هناك ملف web.config في c:\inetpub\wwwrootفإن الإعدادت تطبق على جميع المجلدات الفرعية لهذا المجلد مالم يوجد ملف web.config لمجلد داخلي يحتوي على إعدادت مختلفه.

وهنا سوف نتكلم عن أهم الوسوم في ملف web.config لأنه ليس من الضروري كتابة جميع وسوم هذا الملف في كل مشروع وإنما المهم هو كتابة الوسوم التي تحتاجها فقط ولكن قبل أن نتكلم عن الوسوم بالتفصيل الق نظرة على ملف web.config التالي الذي يحتوي الكثير من هذة الوسوم حتى تعرف كيف شكل هذا الملف وكيفية تنسيق الوسوم فيه:

```
<?xml version="1.0" encoding="windows-1256" ?>

<configuration>
<appSettings>
```

```
<add key ="strConn" value="PROVIDER=Microsoft.Jet.OLEDB.4.0;DATA
SOURCE=C:\DB.mdb" />
</appSettings>

<location path="admin">
<system.web>
<authorization>
<deny users="?" />
</authorization>
</system.web>
</location>

<system.web>
<compilation defaultLanguage="C#" debug="true" />
<customErrors mode="Off"/>

<authentication mode="Forms">
<forms name=".aspxformauth" loginUrl="admin/login.aspx" protection="All" path="/"
timeout="15" />
</authentication>

<authorization>
<allow users="*"/>
</authorization>

<globalization fileEncoding="windows-1256" requestEncoding="windows-1256"
responseEncoding="windows-1256" />

</system.web>

</configuration>
```

وهنا أهم الوسوم التي قد تحتاجها كثيرا:

- Configuration
- appSettings
- system.web
- location
- Compilation
- CustomErrors
- Authentication
- Authorization
- Globalization
- SessionState

Configuration

جميع وسوم ملف web.config تكتب داخل هذا الوسم.

```
<Configuration>
</Configuration>
```

appSettings

هذا الوسم يمثل مكان خاص لتخزين أي قيم خاصة ترغب في إستخدامها في مشروعك وهو يحتوي على خاصيتين هما المفتاح key والقيمة value ويمكن إضافة العديد من القيم بإضافة وسم add لكل قيمة.

فعلى سبيل المثال الكود التالي يقوم بحفظ جملة الربط بقاعدة البيانات : Connection String

```
<appSettings>
<add key ="strConn" value="PROVIDER=Microsoft.Jet.OLEDB.4.0;DATA SOURCE=C:\DB.mdb" />
</appSettings>
```

ولإستعمال قيمة جملة الربط هذة في أي ملف aspx ماعليك الا كتابة التعريف التالي:

```
String strConn= ConfigurationSettings.AppSettings("strConn");
```

وبهذة الطريقة يسهل عليك مثلا تغيير إسم قاعدة البيانات لأن جميع الصفحات ترتبط بجملة ربط واحدة.

System.web

هذا الوسم أيضا تكتب داخلة كثير من الوسوم الأخرى ولكن ليس جميع الوسوم كما في configuration وكذلك ممكن أن يتكرر خلال ملف web.config واحد حسب عدد المجلدات الفرعية التي تحتاج الى كتابة إعدادت خاصة بها.

Location

يمكنك هذا الوسم من إضافة إعدادت معينة لملف أو مجلد معين دون أن تأثر هذة الإعدادات على بقية الملفات أو المجلدات وهو يمكن إذن أن يحتوي على أي من الوسوم الأخرى التي تستخدم لضبط إعدادت لملف أو مجلد معين.

وهذا الكود يستخدم مثلا لحماية المجلد الفرعي admin بحيث لايسمح بدخول أي زائر لأي ملف من ملفات هذا المجلد .

```
<location path="admin">
<system.web>
<authorization>
<deny users="?" />
</authorization>
</system.web>
```

```
</location>
```

Compilation

في هذا الوسم يتم إضافة جميع خصائص ترجمة وتنفيذ الصفحات. وهو يحتوي على عدد من الخصائص مثل defaultLanguageلتحديد نوع اللغة المستخدمة في الصفحات وكذلك ٣ وسوم داخلية هي (compilers) لتحديد مترجم الصفحات و (assemblies) لإضافة ملفات Assemblies معينة و (namespaces) لإضافة مكتبة namespacesمعينة.

```
<compilation defaultLanguage="C#" debug="true">
<compilers>
<compiler language="VB;VBScript" extension=".cls" type="Microsoft.VB.
VBCodeProvider,System" />
<compiler language="C#;Csharp" extension=".cs" type="Microsoft.Csharp.
CsharpCodeProvider,System" />
</compilers>
<assemblies>
<add assembly="ADODB" />
</assemblies>
<namespaces>
<add namespace="System.Web" />
<add namespace="System.Web.UI" />
</namespaces>
</compilation>
```

CustomErrors

هذا الوسم يمكنك من تحديد ظهور رسالة الخطأ أو تحويلة الى أي صفحة أخرى في حالة حدوث أي خطأ في الصفحة ويحتوي هذا الوسم على خاصيتين هما defaultRedirect لتحديد الصفحة التي يتم تحويله لها و mode وتحتوي على أحدى ثلاث قيم هي On وتعني تشغيل هذة الخاصية وتتبع الصفحة المحددة و OFFوتعني إظهار رسالة الخطأ الحقيقية للجميع و RemoteOnly تعني أن يظهر الخطأ الحقيقي فقط لمن يطلب الصفحة من نفس السيرفر أما الذي يطلبها من جهاز آخر فإنه يتم تحويلة الى الصفحة المحددة في defaultRedirect . وكذلك تحتوي على وسم داخلي هو error لتحويلة الى صفحة معينة في حالة خطأ معين مثل أن يكون رمز الخطأ ٤٠٤ أي عدم وجود الصفحة:

```
<customErrors defaultRedirect="errorpage.aspx" mode="RemoteOnly">
<error statusCode="404" redirect="NotFound.htm"/>
</customErrors>
```

Authentication

في هذا الوسم يتم تحديد خصائص الحماية وصفحة الدخول للأشخاص المسموح لهم بالدخول لصفحات الموقع وتحتوي على خاصية واحدة هي mode ويمكن أن تأخذ أحد أربع قيم وهي None أي بدون حماية و Forms أي نموذج

لإدخال الرقم السري و Windows لإستخدام نفس المستخدمين لنظام التشغيل و Passport وهي خدمة من Microsoftيمكن الإشتراك بها .

```
<authentication mode="Forms">
<forms name=".aspxformauth" loginUrl="admin/login.aspx" protection="All" path="/"
timeout="15" />
</authentication>
```

Authorization

وهذا الوسم يمكننا من تحديد الأشخاص المسموح لهم أو الممنوعين من دخول صفحات الموقع ويحتوي على وسمين داخليين هما allow و deny وكل منهم يحتوي على ثلاث خصائص هي users لتحديد المستخدمين المسموح لهم أو الممنوعين ويتم تحديدهم بذكر أسمائهم أو كتابة * وتعني جميع المستخدمين أو ؟ وتعني الزوار ... وخاصية roles لتحديد نوع المستخدمين المسموح لهم أيضا أو الممنوعين ... وخاصية verb وتحدد المستخدمين حسب نوع طلب الصفحة وتحتوي على أربع قيم هي GET و HEAD و POST و DEBUG.

```
<authorization>
<allow roles="Admins" />
<deny users="?" />
</authorization>
```

Globalization

وفي هذا الوسم يتم تحديد الترميز اللغوي ويحتوي على ثلاث خصائص هي fileEncoding وتعني ترميز الصفحات و requestEncoding لترميز الطلب و responseEncoding لترميز الكتابة.

```
<globalization fileEncoding="windows-1256" requestEncoding="windows-1256"
responseEncoding="windows-1256" />
```

SessionState

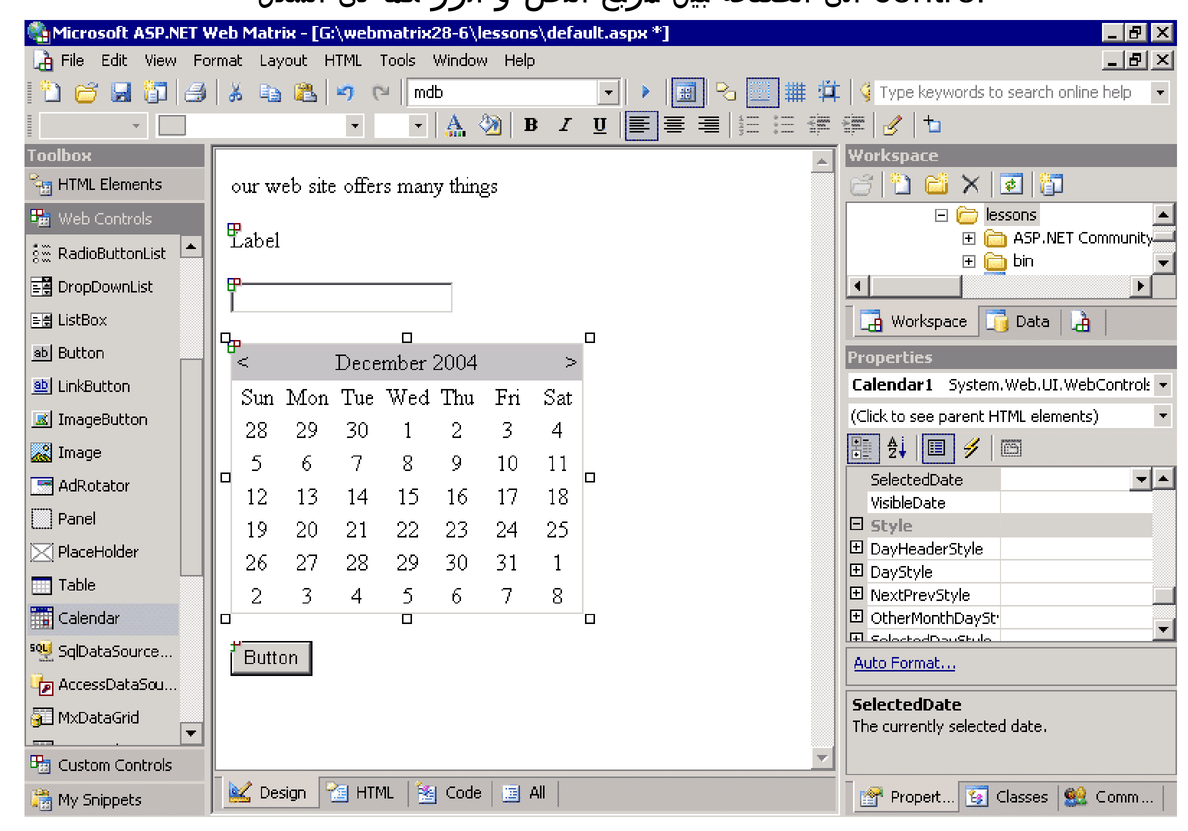
في هذا الوسم يتم تحديد خصائص الجلسة Session وأين يتم حفظها وتحتوي على عدد من الخصائص مثل mode ويمكن أن تحوي أي من هذة الأربع قيم وهي Off وتعني عدم حفظها و InProc وتعني حفظة في الجهاز المحلي و StateServerأي حفظه في السيرفر و SqlServer تعني حفظة في قاعدة بيانات . SqlServer وتحتوي خاصية cookielessأي حفظها في كوكيز أو لا وكذلك خاصية timeout وتعني مدة حفظها وتأخذ قيمة إفتراضية تساوي ٢٠ دقيقة في حال عدم تحديدها.

```
<sessionState mode="Inproc" cookieless="true" timeout="20">
```

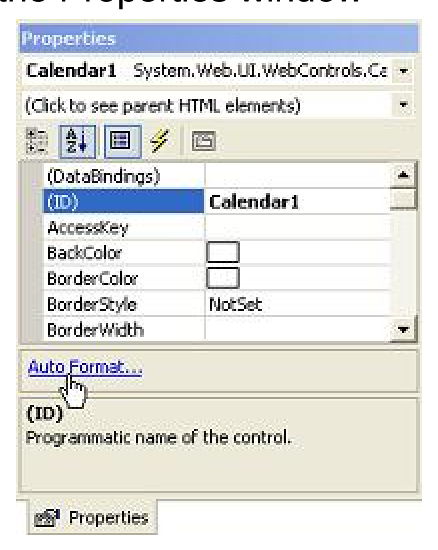
اعتقد انك الان أصبحت متألف مع البرنامج و الكود لحد ما لكن تود ان تعرف أكثر عن كيفية استخدام بقية ال Controls

لذالك سوف نجرب معا Calendar Server Control

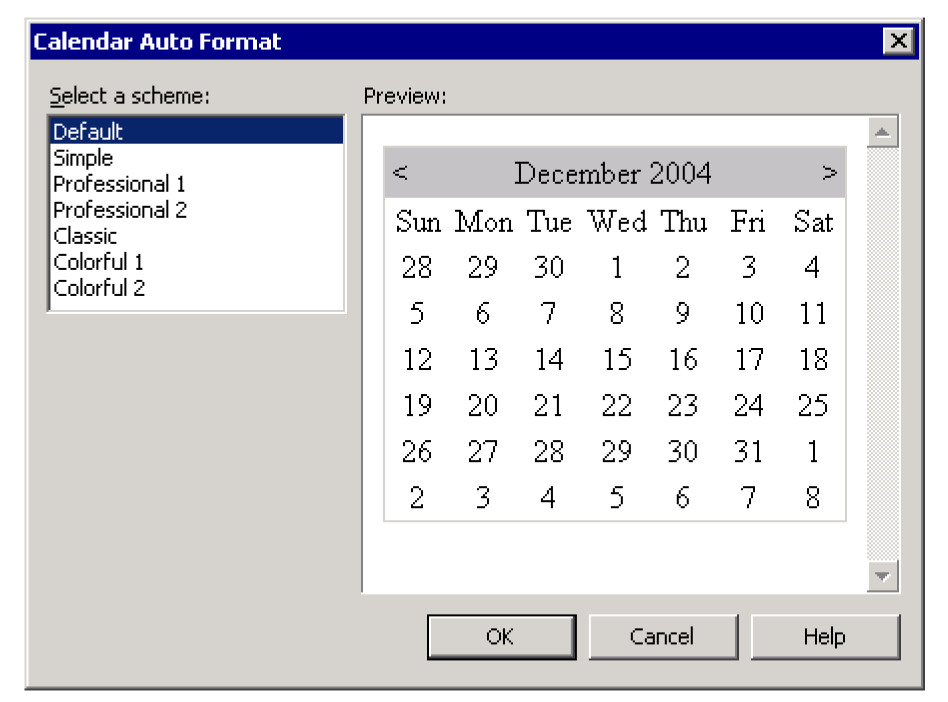
١- افتح الملف myFirstPage.aspx ثم انقر على شاشة Design و اسحب الاداة Calendar control الى الصفحة بين مربع النص و الزر كما فى الشكل



٢- انقر على Calendar بالماوس ثم اضغط على Auto Format الموجودة فى نافذة الخصائص the Properties window الموجودة على اقصى يمين البرنامج



فتظهر نافذة Calender Auto Format التى تتيح لنا تغير اللون و الشكل



انقر على Simple



فتلاحظ تغير الشكل انقر على Ok ثم قم بالتحويل لكود Html لترى الاختلاف

```
<asp:Calendar id="Calendar1" runat="server" BackColor="White" Width="200px"
DayNameFormat="FirstLetter" ForeColor="Black" Height="180px" Font-Size="8pt"
Font-Names="Verdana" BorderColor="#999999" CellPadding="4">
   <TodayDayStyle
      forecolor="Black" backcolor="#CCCCCC">
   </TodayDayStyle>
   <SelectorStyle
      backcolor="#CCCCCC">
   </SelectorStyle>
   <NextPrevStyle
      verticalalign="Bottom">
   </NextPrevStyle>
   <DayHeaderStyle
      font-size="7pt" font-bold="True" backcolor="#CCCCCC">
   </DayHeaderStyle>
   <SelectedDayStyle
```

```
      font-bold="True" forecolor="White" backcolor="#666666">
   </SelectedDayStyle>
   <TitleStyle
      font-bold="True" bordercolor="Black" backcolor="#999999"></TitleStyle>
   <WeekendDayStyle
      backcolor="#FFFFCC">
   </WeekendDayStyle>
   <OtherMonthDayStyle
      forecolor="Gray">
   </OtherMonthDayStyle>
</asp:Calendar>
```

٣- قم بالتغير لوضع Design ثم انقر على Calendar1 مرتين ليظهر شاشة كتابة الكود و اكتب الكود التالى لكى يتغير محتوى اداة النص Label1 بالاختيار الذى يختارة المستخدم من النتيجة Calendar

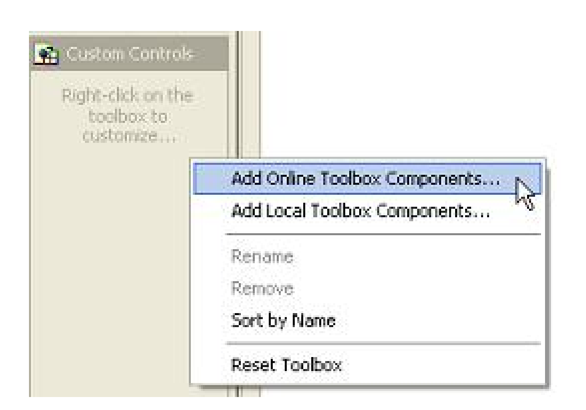
```
void Calendar1_SelectionChanged(object sender, EventArgs e)
{
Label1.Text = "Hello " + TextBox1.Text + " you selected: " +
Calendar1.SelectedDate;
}
```

الان قد تسأل هل يمكن اصنع أداة خاصة بى كالنتيجة السابقة لتقوم بعمل مهام شبيهه بهذة أو أى مهام أخرى ؟ أن تسألك فى مودعة بالضبط فأننا يمكننا صنع الأدوات الخاصة بنا من خلال إتقاننا للغة C# و لكن لماذا نتعب من البداية فهناك آخرين قد يكونوا صنعوا ما تحتاج لة فقط ابحث قبل ان تعمل شئ تعتقد انة جديد لكنة فى الحقيقة سبق عملة و الان تسأل أين تبحث وعن ماذا تبحث ؟

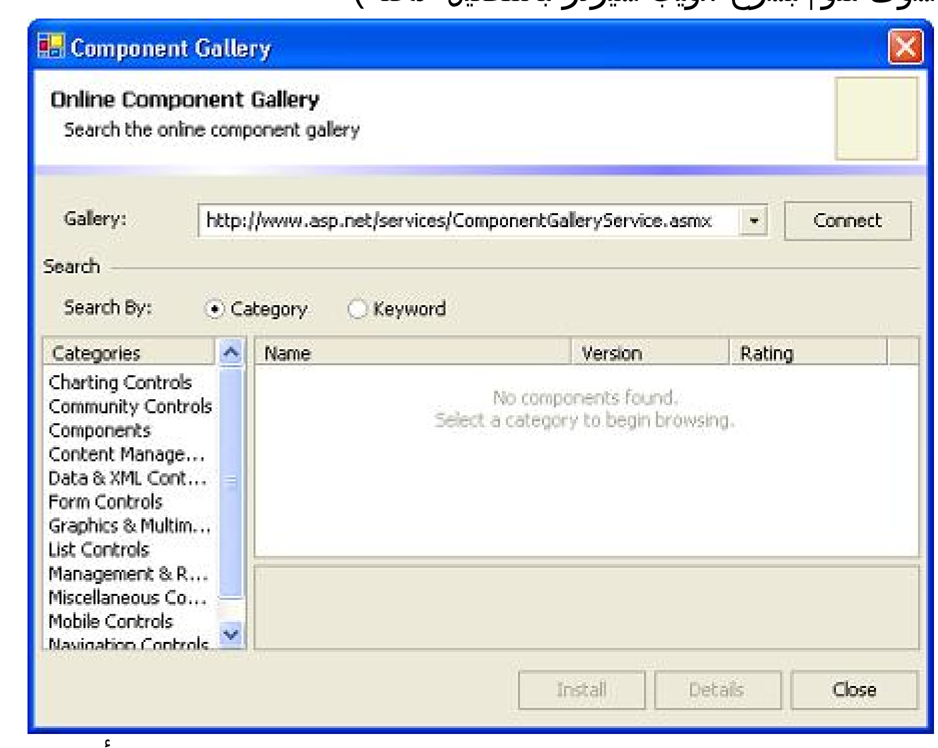
حسنا ان برنامج Web Matrix يمكنك من استخدام ادوات من صنع مبرمجين اخرين و يوفرها لك على الرابط http://www.asp.net/services/componetGalleryService.asmx

و الان اتبع المثال التالى لتعرف الرد على كل الاسئلة التى سبق و ان سألتها

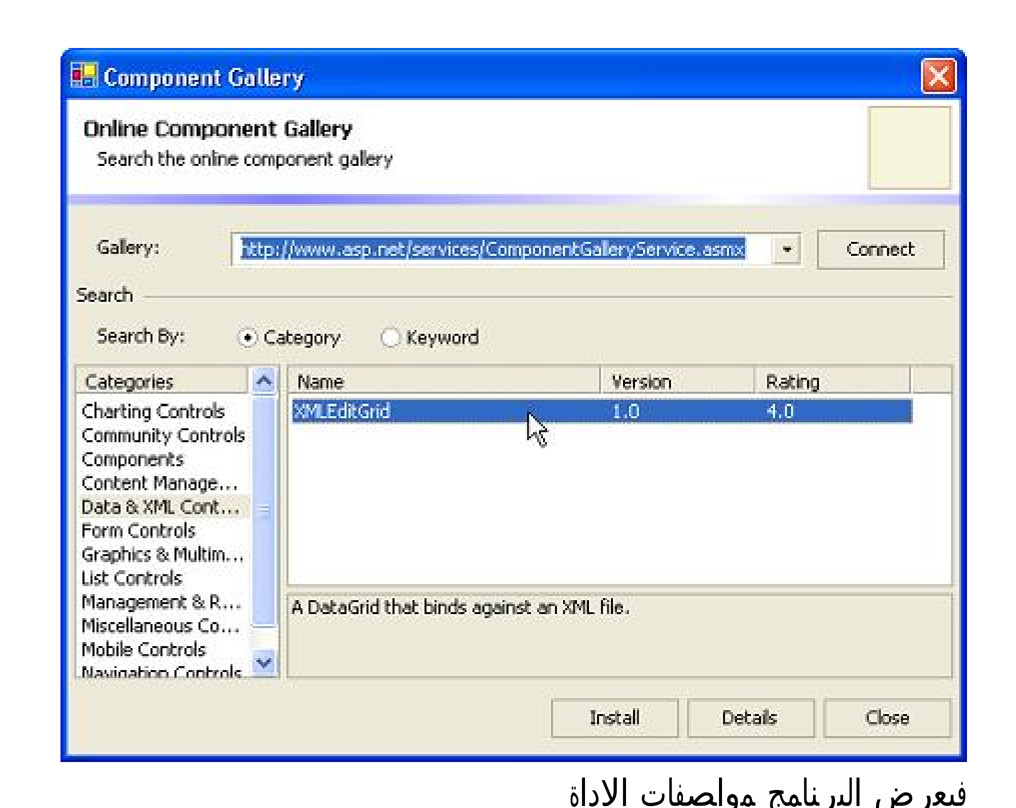
١- قم بعمل صفحة جديدة و ليكن أسمها myFirstCustomPage.aspx ثم انقر على Custom Controls من صندوق الادوات ToolBox ثم قف بالماوس فى المساحة الفارغة الخاصة ب Custom Controls ثم اضغط على الزر الايمن للماوس فتظهر لك قائمة كالتالى :



اختر من القائمة Add Online Toolbox Components ثم يظهر مربع حوار يمكنك من استخدام ويب سيرفز من موقع WWW.asp.net و هى المسئولة عن الاداة الجديدة التى سوف تقوم بأنزالها من الانترنت (لا داعى للقلق سوف نقوم بشرح الويب سيرفز بالتفصيل لاحقا)



اختر من Categories الاداة المناسبة لك ولنفرض مثلا اننى اريد أداة تكون خاصة بعرض ملف Xml داخل صفحتى فى هذة الحالة سوف اختار Data & XML Controls



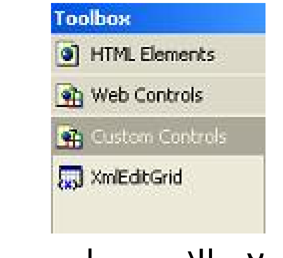
فيعرض البرنامج مواصفات الاداة

ببساطة وظيفتها عرض محتويات ملف Xml داخل DataGrid ثم أنقر بعد ذلك على Instal

الان يظهر رسالة اختر Yes



ترى بعد ذلك الاداة الجديدة موجودة كما فى الشكل



٢- الان نريد ان نجرب الاداة الجديدة و ايضا فرصة لكى نتعامل مع ملف Xml حسنا قم بسحب الاداة XmlEditGrid للصفحة ثم اذهب لقائمة ملف File و انقر على new و اختر من General ما يناسب مشروعنا وهو عمل ملف Xml و ذلك بالاختيار XML File و اختر مكان حفظ الملف على القرص الصلب ثم قم بتسمية الملف myFirstXMLFile.xml

ثم أنقر على Ok .

سوف يفتح ملف Xml امامك قم بأضافة الاكواد التالية لة

```
<?xml version="1.0" encoding="utf-8" ?>
<Orders>
   <Order>
```

```
    <ID>1</ID>
    <OrderDate>4/23/2002</OrderDate>
    <CustomerName>Bob Smith</CustomerName>
  </Order>
  <Order>
    <ID>2</ID>
    <OrderDate>5/12/2002</OrderDate>
    <CustomerName>Sally Jones</CustomerName>
  </Order>
</Orders>
```

الكود السابق هو ببساطة قاعدة بيانات على هيئة ملف xml بة ID هو رقم العميل و OrderDate تاريخ الطلب و Customer Name أسم العميل و كل عميل توضع
بياناته فى وسم (Tag) <Order> و عندما تنتهى من أدخال بياناته تقفل الوسم الخاص بكل بيان فى البداية ثم تقفل الوسم الخاص بالعميل ككل بالوسم </Order>
أما الملف كلة فيبدأ بالوسم <Orders> و ينتهى بالوسم </Orders> و لاحظ ان كل وسم حساس لحالة الأحرف أى أذا كتبت Oreders ثم و انت تقفل الوسم كتبت orders فسيكون هناك خطأ لان حرف الo الصغير لا يساوى حرف الO الكبير .
أضف عملاء أخرين للملف ثم قم بحفظة و أغلاقة .

File
New... Ctrl+N
Open... Ctrl+O
Save Ctrl+S
Save As...
Close
Print... Ctrl+P
Print Preview...
Print Settings...
Recent Files
Exit

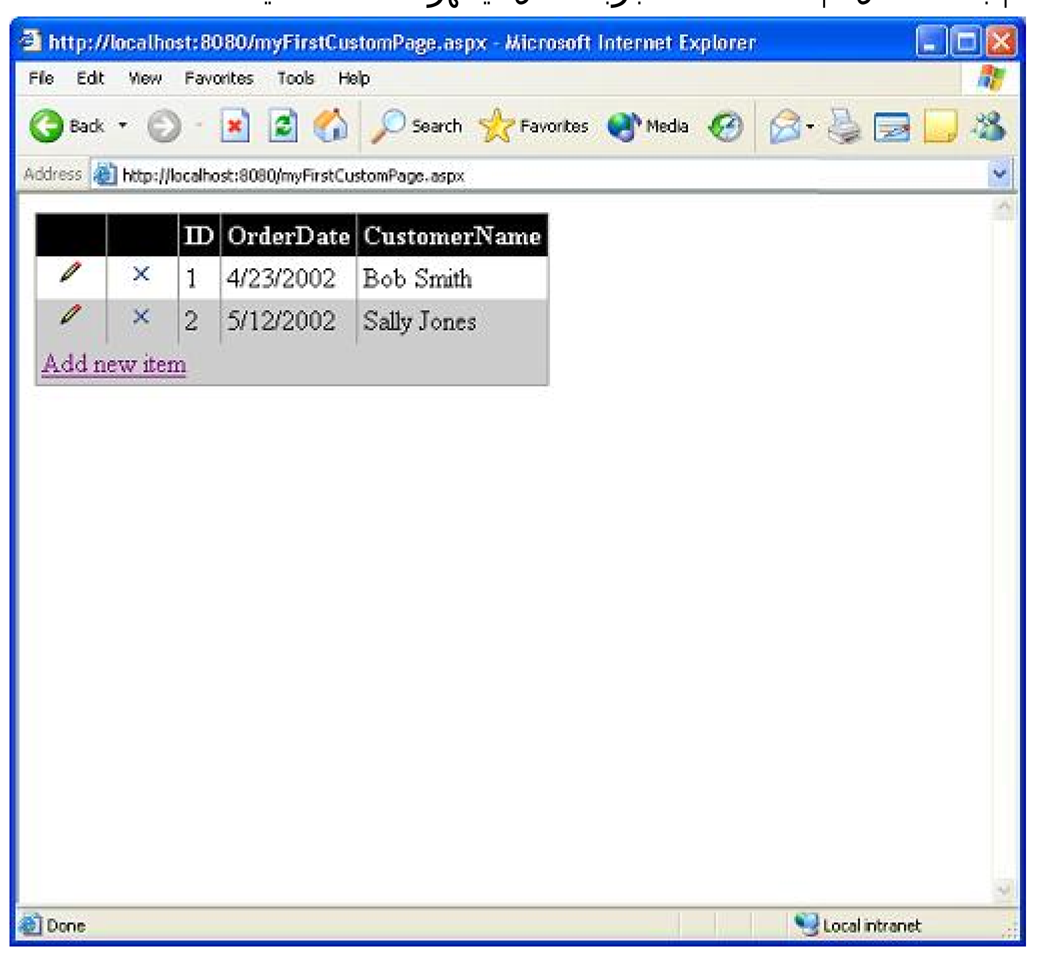
٤- انقر على شاشة Design و انقر على XMLEditGrid control
ثم اخترXmlfile من نافذة الخصائص و قم بكتابة myFirstXMLFile.xml كما فى الشكل

Properties
XmlEditGrid1 Swarren.XmlEditGrid.XmlEditGrid
(Click to see parent HTML elements)

TabIndex	0
ToolTip	
Visible	True
Width	
XmlFile	myFirstXMLFile.xml

Auto Format..., Property Builder...
XmlFile
Xml file to edit.
Properties Classes Community

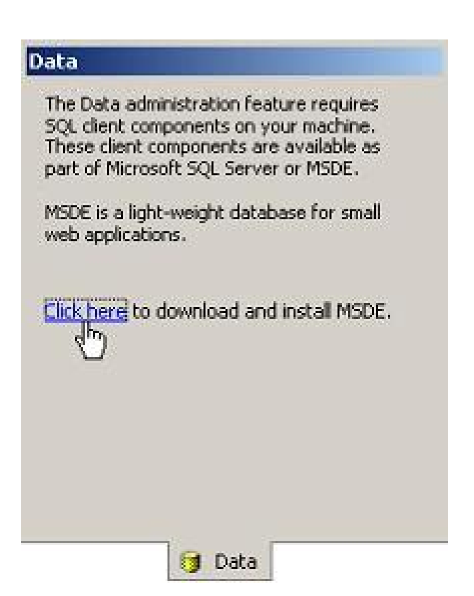
قم بحفظ العمل ثم اضغط F5 لتجربة العمل فيظهر الصفحة التالية



جرب الصفحة و حاول اضافة بيانات جديدة و تعديلها ستجد ان الملف Xml يتم تحديثة تلقائيا باستمرار دون أن تكتب سطر من الكود .

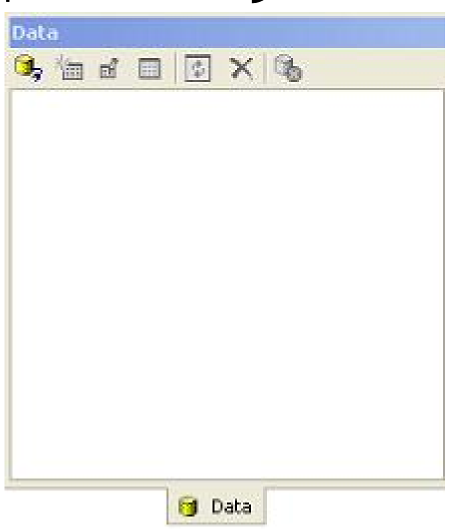
الفصل الثانى التعامل مع قواعد البيانات Database

الربط بقواعد البيانات Connect to a Database Server بداخل البرنامج أمكانية الاتصال بقاعدة بيانات Sql Server 2000 و Access كما يمكننا البرنامج من أنشاء قاعدة البيانات من خلال البرنامج و لكن لكى تستطيع متابعة الشرح يجب أن تقوم بتنزيل Sql Server 2000 على جهازك الشخصى لان اذا لم يكن لديك على الجهاز برنامج SqlServer2000 يقوم البرنامج بأظهار رسالة عندما تضغط على تبويب Data فى Workspace window كالتالى:



و هذة الرسالة تطلب منك تنزيل MSDE و يستحسن تنزيل Sqlserver2000 على الجهاز

أما أذا قمت بتنزيل SqlServer2000 فسوف تظهر لك الشاشة التالية :



تقنية ADO.NET:

صممت تقنية ADO.NET كتطور لتقنية ADO، ولكن الحقيقة أنّهما غير متشابهين.. فبينما تفترض تقنية ADO أن العميل يبقى على اتصال بقاعدة البيانات طوال مدة تعامله معها عبر الشبكة، حيث يحصل على البيانات من أي جدول يريده ويحدثها عبر نفس الاتصال.

هذه الطريقة تبدو عقيمة، ففتح الاتصال طوال الوقت مع قاعدة البيانات عبر الشبكة غير عملي، كما أن إجراء العمليات على الشبكة أبطأ.. هذه ما جعل المبرمجين يستخدمون تقنية ADO بطريقة غريبة، فهم يتصلون بقاعدة البيانات، وينسخون البيانات المطلوبة إلى أجهزتهم، ثم يغلقون الاتصال ويجرون العمليات المطلوبة على البيانات.. فإذا كانت هناك تغييرات يجب حفظها في قاعدة البيانات، فإنّهم يتصلون بقاعدة البيانات مرة أخرى ويحفظون البيانات ثم يغلقون الاتصال.

لذا فقد أضافت ميكروسوفت لتقنية ADO.NET إمكانية التعامل مع البيانات بعد إغلاق الاتصال Disconnected Mode، حيث يتم تحميل البيانات في ذاكرة الجهاز، ليتعامل معها المستخدم على جهازه، وعندما يريد حفظ التغييرات، يتم فتح الاتصال مرة أخرى لنقل البيانات إلى قاعدة البيانات.

ولكن كيف يتعامل العميل مع البيانات على جهازه؟.. إنّه يحتاج لوضعها في تركيب يشبه تركيب قاعدة البيانات الأصلي.. هذا التركيب يسمى "مجموعة البيانات" DataSet، وهي تمكّنك من التعامل مع البيانات بالحذف والإضافة والتعديل، كما تمكّنك من دمج مجموعات من السجلات من أكثر من قاعدة بيانات.. كلّ ذلك دون أن تؤثّر على قاعدة البيانات الأصلية في شيء، إلا إذا أردت حفظ هذه التغييرات لقاعدة البيانات عن طريق إجراء اتصال جديد بها وتحديثها.

إن بإمكانك تحميل كلّ قاعدة البيانات في "مجموعة البيانات" DataSet، وإن كان الأكثر عملية هو أنّك ستحمل الجزء الذي تريد التعامل معه من البيانات، وذلك عن طريق جملة SQL مناسبة.

ويمكنك القول إن "مجموعة البيانات" DataSet هي أشبه ما تكون بمجموعة السجلات RecordSet في تقنيات DAO و RDO و ADO، حيث كانت الجداول ونتائج الاستعلامات تُحفظ فيها، وإن كانت مجموعة البيانات أرقى من مجموعة السجلات، فهي بمثابة صورة مصغّرة من قاعدة البيانات، تحتوي على الجداول والعلاقات فيما بينها والقيود Constraints المفروضة على قيم الحقول، بما في ذلك فرض التكامل المرجعي Referential Integrity بين الجداول.

ولتحميل البيانات في مجموعة البيانات DataSet، يجب عليك أولا أن تتصل بقاعدة البيانات.. إن هذا يتم باستخدام كائن الاتصال Connection object، حيث يسمح لك بتحديد قاعدة البيانات التي ستتعامل معها.. وهناك كائن آخر يقف بين مجموعة البيانات وقاعدة البيانات.. ذلك هو "موصل البيانات" DataAdapter Object، الذي يحتوي على المعلومات اللازمة لنقل البيانات من وإلى قاعدة البيانات.. هذه المعلومات عبارة عن استعلامات SQL اللازمة لقراءة سجلات قاعدة البيانات أو تحديثها أو حذفها أو إضافة بعض الصفوف الجديدة.

إن هذا التخطيط يجعل من مجموعة البيانات أداةً متخصصةً في عرض البيانات مهما كان نوع قاعدة البيانات التي تنتمي إليها، أو كيفية قراءتها وحفظها، فهذه الوظائف من اختصاص موصل البيانات DataAdapter، الذي لا يهتم بدوره بموضع قاعدة البيانات أو كيفية الاتصال بها، فهذه مهمة أداة الاتصال Connection.. وبهذا يمكنك أن تكتب أجزاءً عامة من الكود الذي يتعامل مع مجموعة البيانات، دون أن تحتاج لتغييره تبعا لنوع قاعدة البيانات أو نوع الاستعلام.. كما أن بإمكانك حفظ بيانات مجموعة البيانات في ملفّ XML وإعادة قراءتها منه مرة أخرى.

تسير تقنية ADO.NET و لغة XML متعانقتي اليدين.. إن XML هي طريقة لتمثيل أي بيانات منظّمة، وذلك بتحويلها لنص يعبر عنها.. وبهذا تصلح لغة XML للتعبير عن أي نوع من أنواع البيانات، مثل الجداول والصور وغيرهما.

ورغم أن الملفّات النصية أكبر حجما من الملفّات الثنائية Binary Files، إلا إن الأولى صالحة للتعامل مع أي تطبيق بل مع أي نظام تشغيل.. لهذا فقد صارت لغة XML في السنوات الأخيرة هي أنسب وسيلة لنقل البيانات عبر الإنترنت، وذلك حتّى تتجاوز مشاكل عدم التوافق بين التطبيقات وأنظمة التشغيل المختلفة.. شيء طبيعي إذن أن تستخدم تقنية ADO.NET لغة XML في نقل البيانات بين الخادم والعميل.

ولحسن الحظّ، أنت لا تحتاج لكتابة أو قراءة أي كود XML بنفسك ـ وإن كان ذلك متاحا لك لو احتجته ـ وذلك لأن تقنية ADO.NET تستخدم هذه اللغة كطبقة داخلية، فهي تنتجها وتقرأ البيانات منها بطريقة آلية.

ومن الإمكانيات التي تتيحها لك لغة XML، قدرتك على استخدامها لإنشاء مجموعة بيانات DataSet بدون الاعتماد على أي قاعدة بيانات.. في هذه الحالة ستوضع البيانات في الذاكرة، ولو شئت الاحتفاظ بها، فيمكنك حفظها في ملفّ، ثم إعادة تحميلها مرة أخرى حينما تريد.

كائن الأوامر Command Object:

إن الخصائص SelectCommand و DeleteCommand و InsertCommand و UpdateCommand ما هي إلا كائنات من هذا النوع.

وسنرى كيف ننشئ كائن الأوامر من خلال الكود، ونـضع قيمـا لخصائـصه بأنفـسنا، مثـل خاصـية الاتـصال Connection، وخاصية "نص الأمر" CommandText.. بعد ذلك سنستخدم أحد الوسائل التالية لتنفيذه:

ExecuteReader:

تنفّذ الأمر، وتعيد كائنا من نوع قارئ البيانات DataReader، حيث يمكنك استخدامه لقـراءة النتيجـة سـجلا وراء سجلّ.

ExecuteXMLReader:

تنفّذ الأمر، وتعيد كائنا من النوع XMLDataReader، وهو مشابه لقارئ البيانات.

ExecuteScalar:

تنفّذ الأمر، وتعيد الخانة الموجودة في الصفّ الأول من العمود الأول في الجدول الناتج، وتتجاهل بـاقي الخانـات.. ويمكنك استخدام هذه الوسيلة لتنفيذ دوال التجميع Aggregate Functions.

ExecuteNonQuery:

ينفّذ الأمر، وتعيد فقط عدد السجلات التي تأثّرت بالأمر.. ويمكنك استخدام هذه الوسيلة لتنفيذ أوامر التحديث والحذف والإدراج.

كائن قراءة البيانات DataReader:

إن قارئ البيانات DataReader أسرع بمراحل من مجموعة البيانات، ولكنّه في الوقت ذاته لا يـسمح لـك بتعـديل السجلات، ولا حتّى بالتحرك فيها الخاف، فأنت مجبرٌ على استخدام الوسيلة Read لقراءة السجلات واحدًا تلو الآخر، وفي أثناء ذلك يكون الاتصال بينك وبين قاعدة البيانات مفتوحا.. باختصار: إن قارئ البيانات DataReader يعيد لك نتيجةً توصفُ بأنّها للأمام فقط وللقراءة فقط Forward-only, Read-only Result.

هناك نوعين من قارئ البيانات:

SqlDataReader : خاص بقواعد البيانات Sql server 2000

OleDbDataReader: خاص بالاكسيس

الاتصال بقاعدة البيانات Connecting to a Database

خصائص الامر ConnectionString

Possible Parameters for ConnectionString	
Parameter	**Description**
`Provider=`	The name of the data provider (Jet, SQL, and so on) to use.

	اسم البرنامج الذى تم صنع قاعدة البيانات بواسطتة مثلا (Jet, SQL)
Data Source=	The name of the data source (database) to connect to. اسم قاعدة البيانات
UID=	A valid username to use when connecting to the data source. اسم المستخدم لقاعدة البيانات
PWD=	A password to use when connecting to the data source. كلمة السر الخاصة بقاعدة البيانات
DRIVER=	The name of the database driver to use. This isn't required if a DSN is specified.
SERVER=	The network name of the data source server. اسم السيرفر الخاص بقاعدة البيانات لا يستخدم مع الاكسيس

مثال على ربط صفحة الويب بقواعد البيانات

أولا : الربط مع برنامج Sql server2000 و قاعدة بيانات جاهزة اسمها pups

```
<% @Page language='c#' debug='true' %>
<% @Import namespace='System.Data' %>
<% @Import namespace='System.Data.SqlClient' %>

<SCRIPT runat='server'>

void Page_Load(Object Sender, EventArgs e)
{
SqlConnection cn;
cn = new
SqlConnection("server=localhost;uid=sa;pwd=;database=pubs;");
cn.Open();
Response.Write("Opened connection to " + cn.Database + "<BR>");
Response.Write("SQL Server version " + cn.ServerVersion);
}

</SCRIPT>
```

ثانيا : عمل ربط باستخدام **OLE DB Provider**

```
<% @Page language='c#' debug='true' %>
<% @Import namespace='System.Data' %>

<% @Import namespace='System.Data.OleDb' %>

<SCRIPT runat='server'>

void Page_Load(Object Sender,EventArgs e)
{
OleDbConnection cn;
cn = new OleDbConnection("provider=SQLOLEDB;server=localhost;" +
"uid=sa;pwd=;database=pubs;");
cn.Open();
Response.Write("Opened connection to " + cn.Database + "<BR>");
Response.Write("SQL Server version " + cn.ServerVersion);
}
```

```
</SCRIPT>
```

ثالثا: عمل ربط بالداتا بيس مع تتبع الاخطاء Error Handling

```
<% @Page language='c#' debug='true' %>
<% @Import namespace='System.Data' %>
<% @Import namespace='System.Data.SqlClient' %>

<SCRIPT runat='server'>

void Page_Load(Object Sender, EventArgs e)
{
SqlConnection cn;
cn = new
SqlConnection("server=localhost;uid=sa;pwd=;database=pubs;");
try
{
cn.Open();
Response.Write("Opened connection to " + cn.Database + "<BR>");
Response.Write("SQL Server version " + cn.ServerVersion);
cn.Close();
}
catch(SqlException sx)
{
Response.Write("Connection failed: " + sx.Message);
}
}

</SCRIPT>
```

رابعا: استخدام SqlDataAdapter و DataSet لعرض ناتج استعلام من الداتا بيس

```
<% @Page language='c#' debug='true' %>
<% @Import namespace='System.Data' %>
<% @Import namespace='System.Data.SqlClient' %>

<SCRIPT runat='server'>

void Page_Load(Object Sender,EventArgs e)
{
SqlConnection cn;
SqlDataAdapter da;
DataSet ds;
String strSQL;

strSQL = "SELECT TOP 10 au_fname, au_lname FROM authors";
cn = new
SqlConnection("server=localhost;uid=sa;pwd=;database=pubs;");
da = new SqlDataAdapter(strSQL, cn);
** // بالبيانات العائدة من تنفيذ جملة الاستعلامDataSet نقوم بتحميل ال
ds = new DataSet();
// بمحتويات الداتاسيت SqlDataAdapterنقوم بتحميل
```

```
da.Fill(ds, "Authors");

** //    نعرض البيانات
foreach(DataRow Author in ds.Tables["Authors"].Rows)
{
Response.Write(Author["au_fname"].ToString() + " " +
Author["au_lname"].ToString() + "<BR>");
}

}

</SCRIPT>
```

خامسا: عرض البيانات من خلال DataGrid Control

```
<% @Page language='c#' debug='true' %>
<% @Import namespace='System.Data' %>
<% @Import namespace='System.Data.SqlClient' %>

<HTML>
<SCRIPT runat='server'>

void Page_Load(Object Sender,EventArgs e)
{
SqlConnection cn;
SqlDataAdapter da;
DataSet ds;
String strSQL;

strSQL = "SELECT TOP 10 au_fname, au_lname FROM authors";
cn = new SqlConnection("server=localhost;uid=sa;" +
"pwd=;database=pubs;");
da = new SqlDataAdapter(strSQL, cn);_

// ** Fill DataSet
ds = new DataSet();
da.Fill(ds, "Authors");
// ** Display data
DataGrid1.DataSource = ds.Tables["Authors"].DefaultView;
DataGrid1.DataBind();


}

</SCRIPT>

<BODY>

<ASP:datagrid id='DataGrid1' runat='server' />


</BODY>
```

```
</HTML>
```

ملحوظة يمكن استخدام الوسيلة DataBind() مع ArrayList Object

```
<% @Page language='c#' debug='true' %>
<% @Import namespace='System.Collections' %>

<HTML>
<SCRIPT runat='server'>

void Page_Load(Object Sender,EventArgs e)
{
ArrayList al = new ArrayList();
al.Add("Alaska");
al.Add("Alabama");
al.Add("California");
al.Add("Kentucky");

// ** Display data
DataGrid1.DataSource = al;
DataGrid1.DataBind();
}

</SCRIPT>

<BODY>

<ASP:datagrid id='DataGrid1' runat='server' />

</BODY>
</HTML>
```

سادسا: عمل نموذج للأدخال البيانات فى قاعدة البيانات

```
<% @Page language='c#' debug='true' trace='false' %>
<% @Import namespace='System.Data' %>
<% @Import namespace='System.Data.SqlClient' %>

<SCRIPT runat='server'>
void Page_Load(Object Sender,EventArgs e)
{
if(Page.IsPostBack)
{
SqlConnection cn;
SqlCommand cm;
String strSQL;

cn = new
SqlConnection("server=localhost;uid=sa;pwd=;database=pubs;");
strSQL = "INSERT INTO authors " +
"(au_id, au_fname, au_lname, contract) " +
"VALUES ('" + txtID.Text + "', '" +
txtFirstName.Text + "', '" +
txtLastName.Text + "', '" +
ChkToInt(chkContract) + "')";
```

```
cm = new SqlCommand(strSQL, cn);


// ** Open connection
try
{
cn.Open();
// ** Execute command
cm.ExecuteNonQuery();
}
catch(SqlException sx)
{
Response.Write("Exception Occurred: " + sx.Message);
}
finally
{
if (cn.State == ConnectionState.Open)
cn.Close();
}

// ** Execute command
Trace.Write("Command: " + cm.CommandText);

// ** Clear form for next item
txtID.Text = "";
txtFirstName.Text = "";
txtLastName.Text = "";
chkContract.Checked = false;
}
}

int ChkToInt(CheckBox chk)
{
if(chk.Checked)
return 1;
else
return 0;
}

</SCRIPT>

<html>
<head>
<title>ASP.NET Data Entry</title>
</head>

<body bgcolor="#FFFFFF" text="#000000">
<FORM runat='server'>
<table width="300" border="0">
<tr>
<td>ID: </td>
<td>
<asp:textbox id="txtID" runat='server' />
</td>
</tr>
<tr>
```

```
<td>First Name: </td>
<td>
<asp:textbox id="txtFirstName" runat='server' />
</td>
</tr>
<tr>
<td>Last Name:</td>
<td>
<asp:textbox id="txtLastName" runat='server' />
</td>
</tr>
<tr>
<td>Contract:</td>
<td>
<asp:checkbox id="chkContract" runat='server' />
</td>
</tr>
</table>
<p>
<asp:button id="btnSave" text="Save" runat='server' />
</p>
</FORM>
</body>
</html>
```

التعامل مع الاخطاء الصادرة من الداتا بيس Handling Errors

```
<% @Page language='c#' debug='true' %>
<% @Import namespace='System.Data' %>
<% @Import namespace='System.Data.SqlClient' %>
<HTML>
<SCRIPT runat='server'>
void Page_Load(Object Sender,EventArgs e)
{
SqlConnection cn;
SqlDataAdapter da;
DataSet ds;
String strSQL;
strSQL = "SELECT TOP 10 au_fname  FROM authors"  ;
cn = new
SqlConnection("server=localhost;uid=sa;pwd=;database=pubs;");
da = new SqlDataAdapter(strSQL, cn);
try
{

ds = new DataSet();
da.Fill(ds, "Authors");
DataGrid1.DataSource = ds.Tables["Authors"].DefaultView;
DataGrid1.DataBind();
}
catch(SqlException sx)
{
foreach(SqlError se in sx.Errors)
{
Response.Write("SQL Error: " + se.Message + "<BR>");
```

```
}
}

</SCRIPT>

<BODY>
<ASP:datagrid id='DataGrid1' runat='server' />

</BODY>
</HTML>
```

الأتصال بقاعدة بيانات أكسيسAccess من خلال الكود

لذلك فى البداية سوف نقوم بعمل قاعدة بيانات ببرنامج الاكسيس نذهب للقرص الصلب فى المجلد G:\webmatrix28-6\lessons

نضع به قاعدة بيانات باسم db من نوع أكسيس بها جدول واحد إسمه Categories و به ٣ حقول كالتالي:

1 : id هو المفتاح الوحيد PrimaryKey من نوعAutoNumber

2 : Category من نوع Text

3 : Active من نوع Boolean و هي بالأكسيسYes/No

نقوم بكتابة عدد من الأقسام Categories في الجدول ثم نحفظ البيانات

و نغلق ال MS Access و نتوجه للعمل برمجيا مع ال.MS WebMatrix

و نقوم بعمل صفحة جديدة و ليكن اسمها myfirstdatabase.aspx

١- انقر على تبويب All و قم بمسح كل المكتوب من الاصل بة و الصق بة الكود التالى

```
<%@ Page Language="C#" Debug="True" %>
<%@ import Namespace="System.Data" %>
<%@ import Namespace="System.Data.OleDb" %>
<script runat="server">

void Page_Load()
{
string prov="Provider=Microsoft.Jet.OleDb.4.0;Data Source=" + Server.MapPath("db.mdb");
string SQL="SELECT * FROM Categories where active=True";
OleDbConnection cn = new OleDbConnection(prov);
OleDbDataAdapter da = new OleDbDataAdapter(SQL,cn);
DataSet ds = new DataSet();


cn.Open();

da.Fill(ds,"categories");
int ttlRecords = ds.Tables["categories"].Rows.Count;

Response.Write("<b>The Library has : ( " + ttlRecords.ToString() + " ) Categories.<BR></b>");

for(int i=0;i<ttlRecords;i++)
```

```
{
object catName = ds.Tables["categories"].Rows["category"];
Response.Write(i+1 + "<i>- Category :</i> <b>" + catName + "</b><BR>");
}
cn.Close();

Response.Write("<a href=./datareader.aspx>DataReader Example</a>");

}

</script>
<html dir="rtl">
<head>
</head>
<body>
<form runat="server">
<!–Insert content here →
</form>
</body>
</html>
```

كما نرى قمنا بإدراج NameSpaces من أجل التعامل مع البيانات و استخدمنا System.Data لأنها تحوي على ال DataSet و استخدمنا ال DataAdapter مثلا والذي يستوجب تضمين system.Data.OleDb حيث سنتعامل مع قاعدة بيانات أكسيس.

في المثال التالي توضيح للموجهات:

```
<%@ Page Language="C#" Debug="True" %>
<%@ import Namespace="System.Data" %>
<%@ import Namespace="System.Data.OleDb" %>
```

نلاحظ أنه في أول سطر قمت بإضافة Debug و جعلت قيمتها True و الهدف من ذلك الحصول على نوعية الخطأ عند حدوثه في مرحلة التنفيذ. Compilation

بعد ذلك قمت بكتابة الحدث PageLoad كالتالي:

```
void Page_Load()
{
}
```

و لاحظ ان بدونه لن تعمل الصفحة من حيث ترجمة الأكواد لو كتبناها مباشرة.

بعد ذلك كتبنا الكود الخاص بالإتصال Connection وال DataAdapter و ال dataset كما هو موضح هنا:

```
string prov="Provider=Microsoft.Jet.OleDb.4.0;Data Source=" + Server.MapPath("db.mdb");
string SQL="SELECT * FROM Categories where active=True";
OleDbConnection cn = new OleDbConnection(prov);
OleDbDataAdapter da = new OleDbDataAdapter(SQL,cn);
DataSet ds = new DataSet();
```

نرى أننا عرفنا متغير نصي String يحمل الإسم prov ليحمل بيانات المزود للبيانات و هي قاعدة بيانات أكسيس ذات الإسم db.mdbثم عرفنا متغير نصي أخر SQL به جملة الإستعلام الخاصة بالتنفيذ بعدها عرفنا كائن الإتصال و مررنا له بيانات المزود

ثم أنشأنا DataAdapter و حددنا له جملة الإستعلام و الإتصال و بعدها أنشأنا ال DataSet لتكون جاهزة للتعبئة بالبيانات.

قلنا يلزم لتعبئة ال DataSet إستدعاء الطريقة Fill الخاصة بال DataAdapter و هذا ما فعلناه هنا:

```
cn.Open();

da.Fill(ds,"categories");
int ttlRecords = ds.Tables["categories"].Rows.Count;

Response.Write("<b>The as4web Library has : ( " + ttlRecords.ToString() + " ) Categories.<BR></b>");

for(int i=0;i<ttlRecords;i++)
{
object catName = ds.Tables["categories"].Rows["category"];
Response.Write(i+1 + "<i>- Category :</i> <b>" + catName + "</b><BR>");
}
cn.Close();



حيث قمنا بتعبئته// بجدول أسميناه categories.
أيضا قمنا بتعريف متغير رقمي// ttlRecords و أسندنا له عدد السجلات الموجودة في الجدول كالتالي:

int ttlRecords = ds.Tables["categories"].Rows.Count;

Response.Write("<b>the Library has : ( " + ttlRecords.ToString() + " ) Categories.<BR></b>");

//The result :
The as4web Library has (5) Categories.
```

ثم قمت بعمل دوارة او جملة تكرار Loop من نوع For Loop تدور على حسب السجلات الموجودة في الجدول و في كل مرة تعرض إسم الحقل الخاص بإسم القسم Category كما هو موضح هنا:

```
for(int i=0;i<ttlRecords;i++)
{
object catName = ds.Tables["categories"].Rows["category"];
Response.Write(i+1 + "<i>- Category :</i> <b>" + catName + "</b><BR>");
}
```

من أجل قليل من التوضيح قمت بتعريف متغير Object أسميته CatName داخل الدوارة لأنه سوف يتغير في كل مرة أي خلال الإنتقال بين السجلات و هو يحوي محتوى الحقل Category في السجل و سيكون الناتج على هذا الشكل:

```
The as4web Library has : ( 5 ) Categories.
1- Category : ASP
2- Category : VB.NET
3- Category : C#.NET
4- Category : ADO.NET
5- Category : ASP.NET
```

لنرى هذا التتبع للكود ربما استوعبنا أكثر:

```
object catName = ds.Tables["categories"].Rows[0]["category"];
Response.Write(catName );

// Result : ASP


object catName = ds.Tables["categories"].Rows[2]["category"];
Response.Write(catName );

// Result : C#.NET


object isActive = ds.Tables["categories"].Rows[0]["active"];
Response.Write(isActive );

// Result : True
```

أتمنى أن طريقة الوصول للحقول قد وصلت لكم الان

مثال أخر:

سوف نقوم بعمل نفس ما قمنا به أي عرض الأقسام كما في الطريقة السابقة و لكن باستخدام ال dataReader و الذي يفضل إستخدامه في حالة العرض بسبب سرعته مقارنة بما سبق.

ها هو الكود:

```
<%@ Page Language="C#" Debug="True" %>
<%@ import Namespace="System.Data.OleDb" %>
<script runat="server">

void Page_Load()
{
string prov="Provider=Microsoft.Jet.OleDb.4.0;Data Source=" + Server.MapPath("db.mdb");
string SQL="SELECT * FROM Categories";
OleDbConnection cn = new OleDbConnection(prov);
OleDbCommand cmd = new OleDbCommand(SQL,cn);


cn.Open();


OleDbDataReader reader = cmd.ExecuteReader();

while(reader.Read())
{
Response.Write(reader["category"].ToString() + "<BR>");
}

cn.Close();
}

</script>
<html>
```

```
<head>
</head>
<body>
<form runat="server">
<!–Insert content here →
</form>
</body>
</html>
```

ما هي الDataView ؟

هي ناتج للأوامر معينة توجه نحو ال DataTable الموجود داخل قطعة الذاكرة ال DataSet و لها عدة فوائد مثل ترتيب البيانات المخزنة داخل ال DataTable أو فلترتها "فكرة شرط البيانات" أو البحث عن بيانات معينة و لعل فائدة ال DataView تكمن في أنها تستخرج البيانات من ال DataSet مباشرة بعد إجراء عملية معينة عليها مما ذكرنا و بالتالي لا تشكل ضغطا على السيرفر و توفر سرعة في عرض البيانات و بالتالي تؤثر على عمل التطبيق.

كيف يتم إنشاء الDataView ؟

يتم إنشاؤها عن طريق تمرير إسم ال DataTable إلى ال Constructor الخاص بال DataView كالتالي:

CODE

```
dv=new DataView("myDataTable");
//أو هكذا
dv=new DataView();
dv=ds.Tables["myDataTable"].DefaultView;
```

طبعا من أجل التعامل مع ال DataView يجب تضمين فضاء الأسماء System.Data و هو نفسه الذي يحتوي على ال DataSet.

مثال برمجي:

لنفرض أننا نريد أن نعرض الدروس التابعة للقسم ذو الرقم CatID = 5 فماذا نفعل؟

سوف يقول أحدهم بكل بساطة و بدون ال DataView سوف نستخدم جملة إستعلام SQL مشروطة و انتهى الأمر .. نعم كلام صحيح و منطقي و لكن ماذا لو كان هناك تسلسل في عرض البيانات في نفس الصفحة ؟ هل ستعمل أكثر من جملة إستعلام؟؟

دعوني أكتب المثال أولا ثم نتناقش:

CODE

```
<%@ Page Language="C#" %>
<%@ Import Namespace="System.Data"%>
<%@ Import Namespace="System.Data.OleDb"%>
<script runat="server">
private void Page_Load(){
string prov="Provider=Microsoft.Jet.OleDb.4.0;Data Source=" + Server.MapPath("db.mdb");
string SQL="SELECT * FROM Lessons";
OleDbConnection cn = new OleDbConnection(prov);
OleDbDataAdapter da = new OleDbDataAdapter(SQL,cn);
DataSet ds = new DataSet();
DataView dv = new DataView();



da.Fill(ds,"Lessons");
dv=ds.Tables["Lessons"].DefaultView;
dv.RowFilter="CatID=5";

dg.DataSource=dv;
dg.DataBind();

}

</script>
<html>
<head>
</head>
<body>
<form runat="server">
<asp:DataGrid id="dg" runat="server"/>
</form>
</body>
</html>
```

استخدمت DataGrid تقوم بعرض الأسئلة.

استخدمنا قاعدة بيانات من نوع أكسيس و أنشأنا إتصال مع قاعدة البيانات ثم أنشأنا DataSet و وضعنا بها جدول الدروس و من ثم إستخدمنا ال DataView و أقصد هنا أحد خصائصها و هي فلترة البيانات حيث جعلناها تستخرج فقط الدروس التي تنتمي للتصنيف العام رقم ٥ هكذا:

CODE

```
dv=ds.Tables["Lessons"].DefaultView;
dv.RowFilter="CatID=5";
```

ثم جعلنا مصدر البيانات في ال DataGrid هو ال DataView التي أنشأناها و هي dv ثم قمنا بإستخدام الطريقة Bind من أجل ربط البيانات.

هل يوجد مثال أو فكرة على نقطة إذا كان هناك عدد من ال DataGrid و كل واحدة تستخرج دروسا معينة؟؟

طبعا يوجد و سأعطيكم الفكرة..

مثلا لنفرض أنه عندنا خمس أقسام للبرمجة ولكل واحد عدد من الدروس و نود أن نظهر القسم و تحته الدروس داخل DataGrid فكيف نفعل ذلك؟

بداية نحتاج لدالة تقوم بإسترجاع البيانات على شكل Data View كالتالي:

CODE

```
private DataView bindDg(object catID)
{

dv.RowFilter="catRef="+ catID;
return dv;
}
```

و لنفرض أننا استخدمنا Repeater لعرض الأسماء الخاصة بالأقسام و بداخله DataGrid ستعرض الدروس فماذا نفعل ؟

سنكتب التالي:

CODE

```
<asp:Repeater id="rp" runat="server" EnableViewState="False">
<ItemTemplate>
:: <%# DataBinder.Eval(Container.DataItem,"category")%> ::  <br>
<asp:DataGrid id="dg" runat="server" AutoGenerateColumns="False"  DataSource='<%#
bindDg(DataBinder.Eval(Container.DataItem,"catID"))%>'>
<columns>
..........
</columns>
```

```
</asp:DataGrid>

</itemTemplate>
</asp:Repeater>
```

نرى أنه في منطقة ال DataGrid حددنا ال DataSource و هي عبارة بيانات مسترجعة من دالة bindDg و التي يمرر لها رقم القسم الخاص بها و الدالة تقوم بإستقبال الرقم و عمل فلترة لل DataTable الخاص بالدروس و الموجود داخل ال DataSet .

<u>لكن كيف تقسم النتائج المعروضة داخلdatagrid</u>

CODE

```
<%@ Page Language="C#"%>
<%@ Import Namespace="System.Data.SqlClient" %>
<%@ Import Namespace="System.Data" %>

<script runat=server>
SqlConnection conNorthwind;
string  strSelect;
int intStartIndex;
int intEndIndex;

void Page_Load(Object sender , EventArgs e)
{
SqlCommand cmdSelect;

conNorthwind = new SqlConnection( @"Server=localhost;Integrated
Security=SSPI;Database=Northwind" );
if (! IsPostBack )
{
// Get  Total Pages
strSelect = "Select Count(*) From Products";
cmdSelect = new SqlCommand( strSelect, conNorthwind );
conNorthwind.Open();
dgrdProducts.VirtualItemCount = ( (int)cmdSelect.ExecuteScalar() / dgrdProducts.PageSize );
conNorthwind.Close();
BindDataGrid();
}
}

void BindDataGrid ()
{
SqlDataAdapter dadProducts;
DataSet dstProducts;

intEndIndex = intStartIndex + dgrdProducts.PageSize;
strSelect = "Select * From Products Where ProductID > @startIndex And ProductID <= @endIndex Order
By ProductID";
dadProducts = new SqlDataAdapter( strSelect, conNorthwind );
dadProducts.SelectCommand.Parameters.Add( "@startIndex", intStartIndex );
dadProducts.SelectCommand.Parameters.Add( "@endIndex", intEndIndex );
dstProducts = new DataSet();
dadProducts.Fill( dstProducts );

dgrdProducts.DataSource = dstProducts;
dgrdProducts.DataBind();
}
```

```
void dgrdProducts_PageIndexChanged( object s, DataGridPageChangedEventArgs e ) {
intStartIndex = ( e.NewPageIndex * dgrdProducts.PageSize );
dgrdProducts.CurrentPageIndex = e.NewPageIndex;
BindDataGrid();
}

</Script>

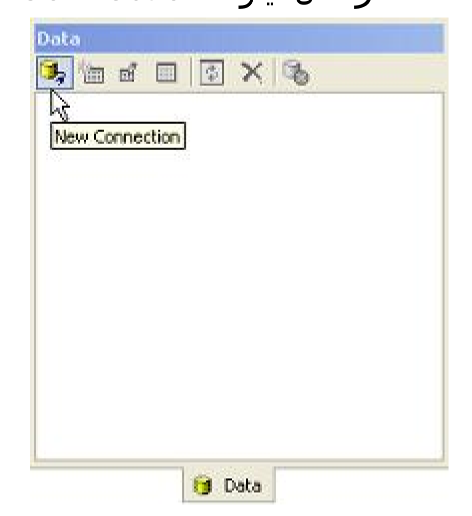
<html>
<head><title>DataGridCustomPaging.aspx</title></head>
<body>
<form Runat="Server">

<asp:DataGrid
ID="dgrdProducts"
AllowPaging="True"
AllowCustomPaging="True"
PageSize="3"
OnPageIndexChanged="dgrdProducts_PageIndexChanged"
PagerStyle-Mode="NumericPages"
CellPadding="3"
Runat="Server" />

</form>
</body>
</html>
```

الان لكى نقوم بالاتصال بقاعدة بيانات SqlServer2000 من خلال البرنامج اتبع التالى :

١- أنقر على ايقونة New Connection

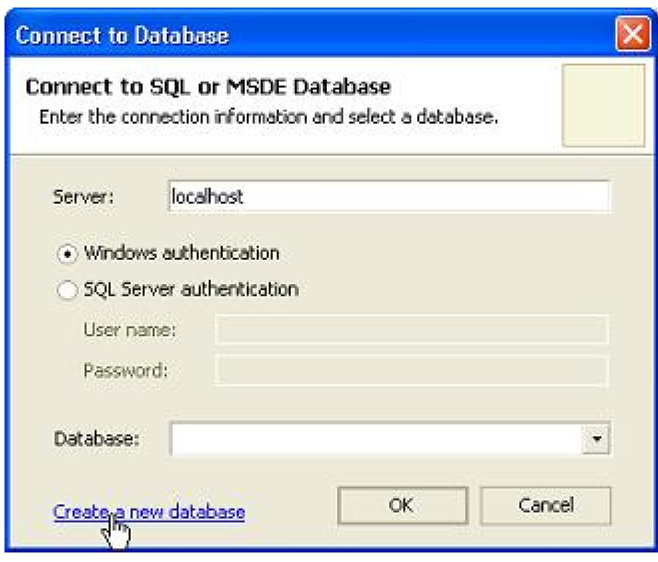


٢- يظهر لك مربع حوار Connect to Database أكتب localhost فى مربع Server و هذا فى حالة قاعدة البيانات الموجودة على جهازك الشخصى أما أذا كانت قاعدة البيانات على شبكة الانترنت فيمكنك كتابة المسار الذى حصلت علية من شركة الاستضافة .

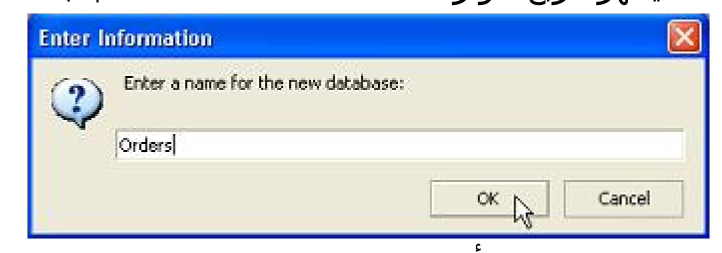
٣- اختر Windows authentication فى حالة أذا كنت تتصل بقاعدة البيانات من خلال تصريح الدخول للنظام الخاص بالويندوز أما اذا كنت اخترت اسم مستخدم و كلمة مرور للنظام عندما قمت بتنزيل Sqlserver2000 فيجب عليك أن تقوم بكتابتهم فى مربعى User name and Password .

اذا كان لديك قاعدة بيانات جاهزة تريد الاتصال بها فيمكنك النقر على السهم بجوار مربع

Database فتظهر قواعد البيانات التى سبق عملها أما أذا كنت ترغب فى عمل قاعدة بيانات جديدة أنقر على Create a new database كما بالشكل التالى :



٤- يظهر مربع حوار Enter Information اكتب بداخلة Orders كأسم لقاعدة البيانات

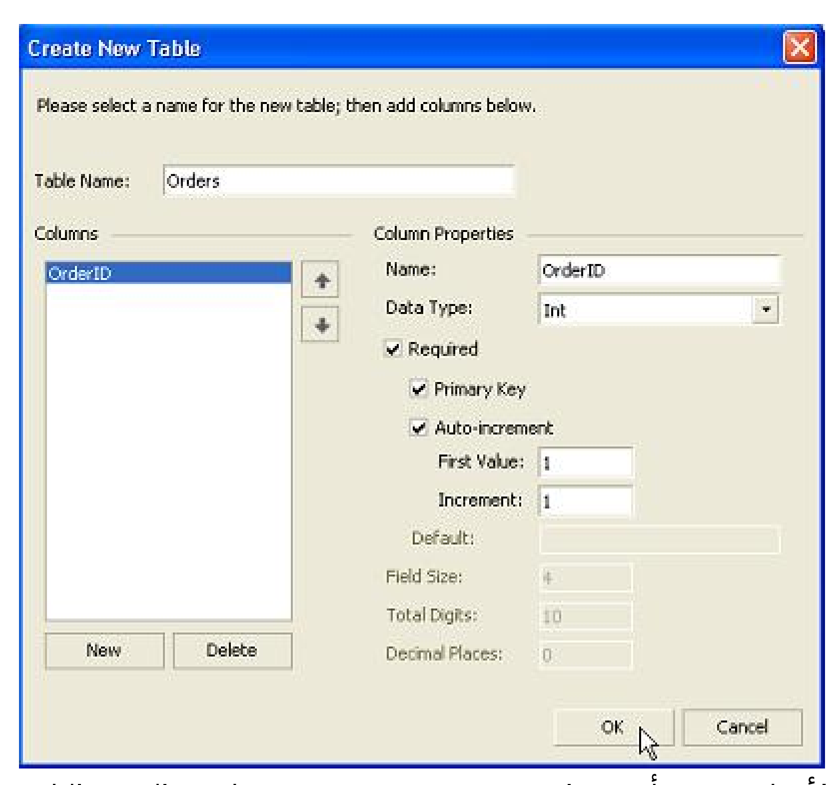


٥- يقوم البرنامج بأنشاء قاعدة البيانات و تلاحظ وجود تكوين شجرى فى نافذة Workspace و رأس الشجرة هو localhost.Orders و الفروع هى Tablesو الاخرى Stored Procedures

٦- قم بالنقر على Tables ثم انقر على New Item فى شريط الادوات



٦- يظهر مربع حوار Create New Table اكتب Orders فى خانة Table Name انقر على New فى اسفل الجزء الخاص ب Columns ثم اكتب OrderID فى خانة Name فى المساحة الخاصة ب Column Properties ثم اختر نوع الحقل Int من القائمة المنسدلة Data Type ثم انقر على Required ثم انقر على Primary Key أى مفتاح رئيسي للجدول ثم انقر على Auto-increment معناها ترقيم تلقائي .



٧- انقر على New لأضافة عمود أخر و اسمة OrderDate و خصائص العمود الثانى ستكون كالتالى :

- Name: OrderDate
- Data Type: DateTime
- Required: True (checked)
- Primary Key: False (unchecked)

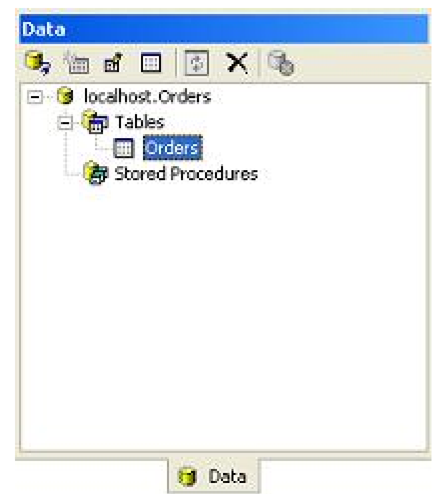
٨- اضف حقل ثالث للجدول orders و سيكون اسم الحقل CustomerName بالخصائص التالية :

- Name: CustomerName
- Data Type: VarChar
- Required: True (checked)
- Primary Key: False (unchecked)
- Field Size: 50 (default)

٩- انقر على Ok لحفظ الجدول داخل قاعدة البيانات



-١٠ يظهر الجدول Orders كفرع من فروع الجداول كما بالشكل :



٧ قم بالنقر على Tables ثم انقر على New Item فى شريط الادوات يظهر مربع حوار Create New Table اكتب OrderDetails فى خانة Table Name ثم أضف مفتاح رئيسى للجدول OrderDetails و ليكن اسمة OrderDetailID بالخصائص التالية:

- Name: OrderDetailID
- Data Type: Int
- Required: True (checked)
- Primary Key: True (checked)
- Auto-increment: True (checked)

اضف عمود جديد OrderID وخصائصة

- Name: OrderID
- Data Type: Int
- Required: True (checked)
- Primary Key: False (unchecked)
- Auto-increment: False (unchecked)

اضف عمود ثالث ProductName و خصائصة

- Name: ProductName
- Data Type: VarChar
- Required: True (checked)
- Primary Key: False (unchecked)
- Field Size: 50 (default)

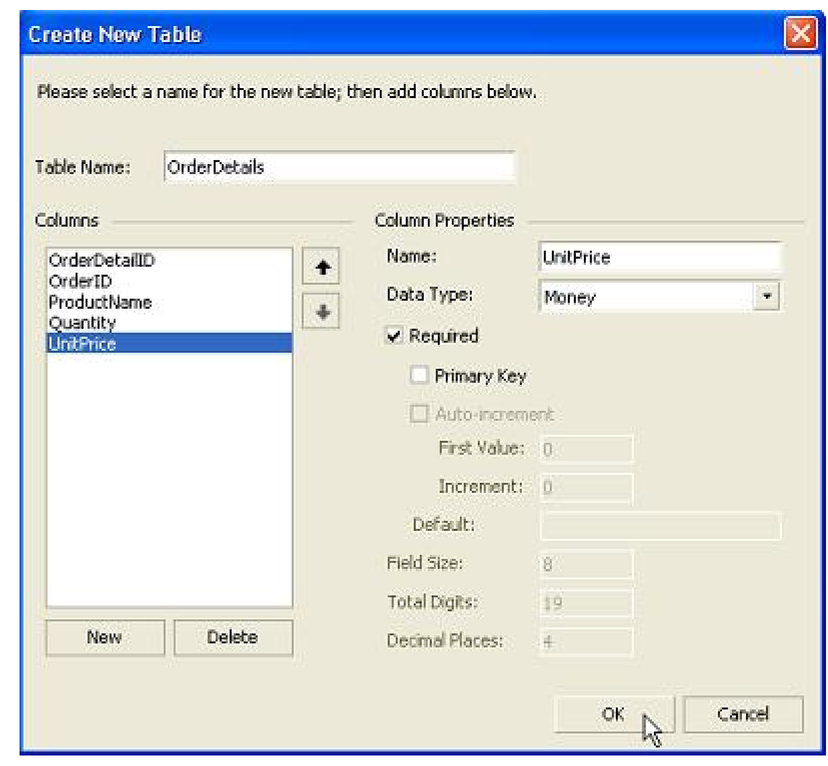
اضف عمود رابع Quantity و خصائصة

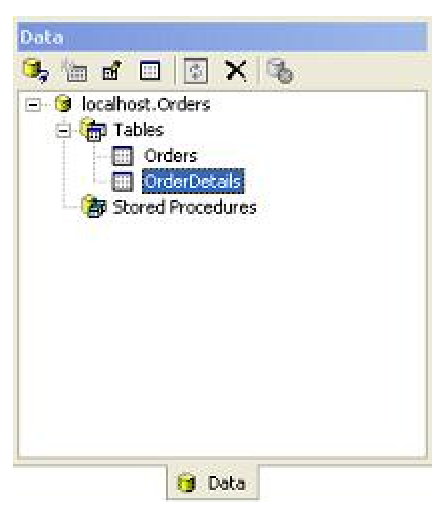
- Name: Quantity
- Data Type: Int
- Required: True (checked)
- Primary Key: False (unchecked)
- Auto-increment: False (unchecked)

اضف عمود خامس UnitPrice و خصائصة

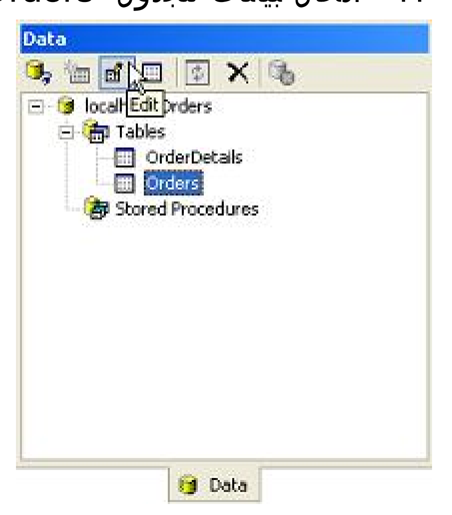
- Name: UnitPrice
- Data Type: Money
- Required: True (checked)
- Primary Key: False (unchecked)

ثم انقر على ok لحفظ الجدول .

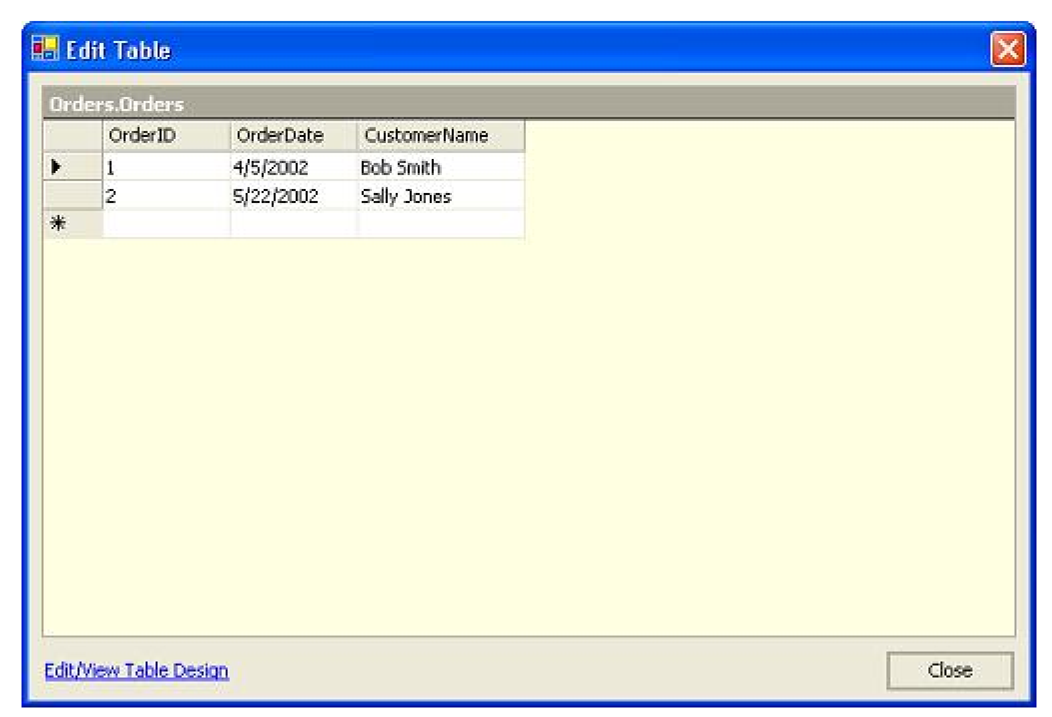




١٢- ادخال بيانات للجدول Orders و للجدول OrderDetails و ذلك من خلال الضغط على Edit



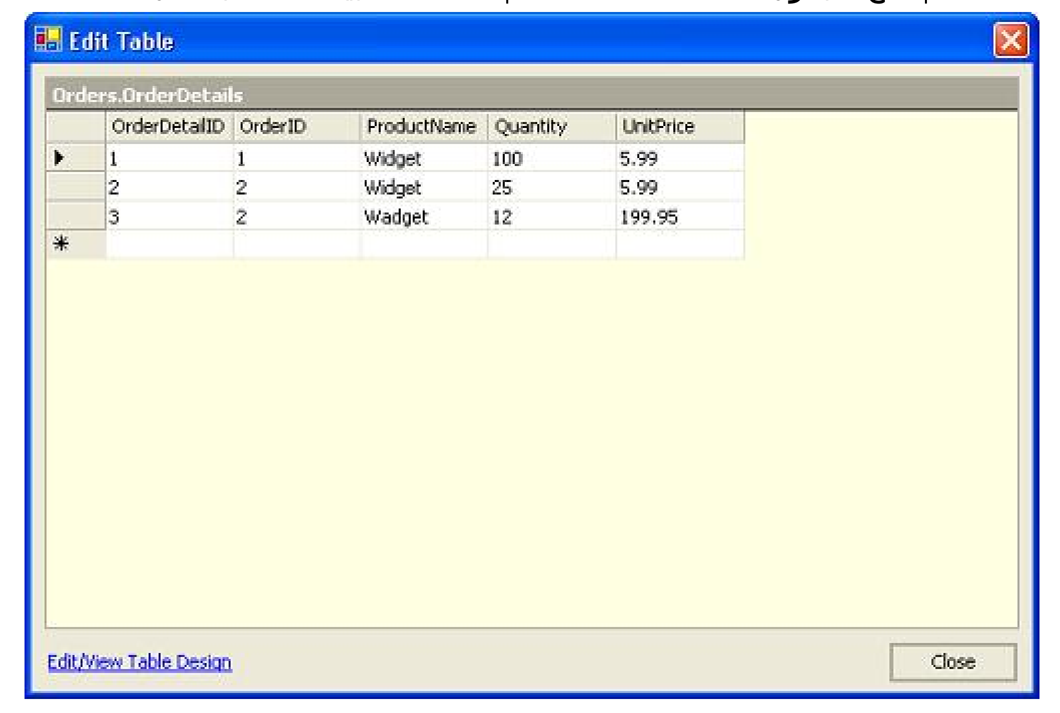
او بالنقر مرتين على الجدول Orders فى نافذة Data ثم قم بأدخال البيانات داخل الجدول كما بالشكل



Orders.Orders

OrderID	OrderDate	CustomerName
1	4/5/2002	Bob Smith
2	5/22/2002	Sally Jones

ثم انقر على Close ثم افتح الجدول OrderDetails ثم اضف لة البيانات كما بالشكل



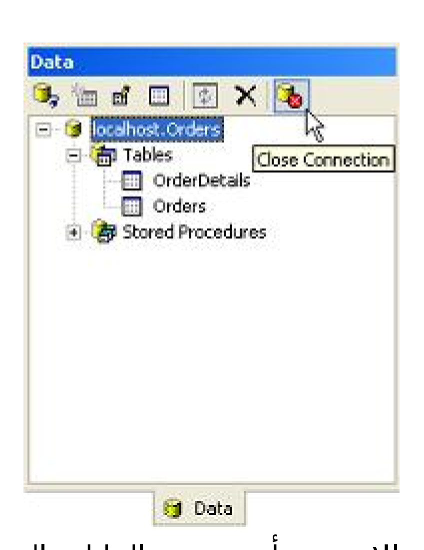
Orders.OrderDetails

OrderDetailID	OrderID	ProductName	Quantity	UnitPrice
1	1	Widget	100	5.99
2	2	Widget	25	5.99
3	2	Wadget	12	199.95

ثم أنقر على Close.

١٣-للأغلاق الاتصال بقاعدة البيانات انقر على (localhost.Orders)

ثم انقر على Close Connection فى نافذة Data

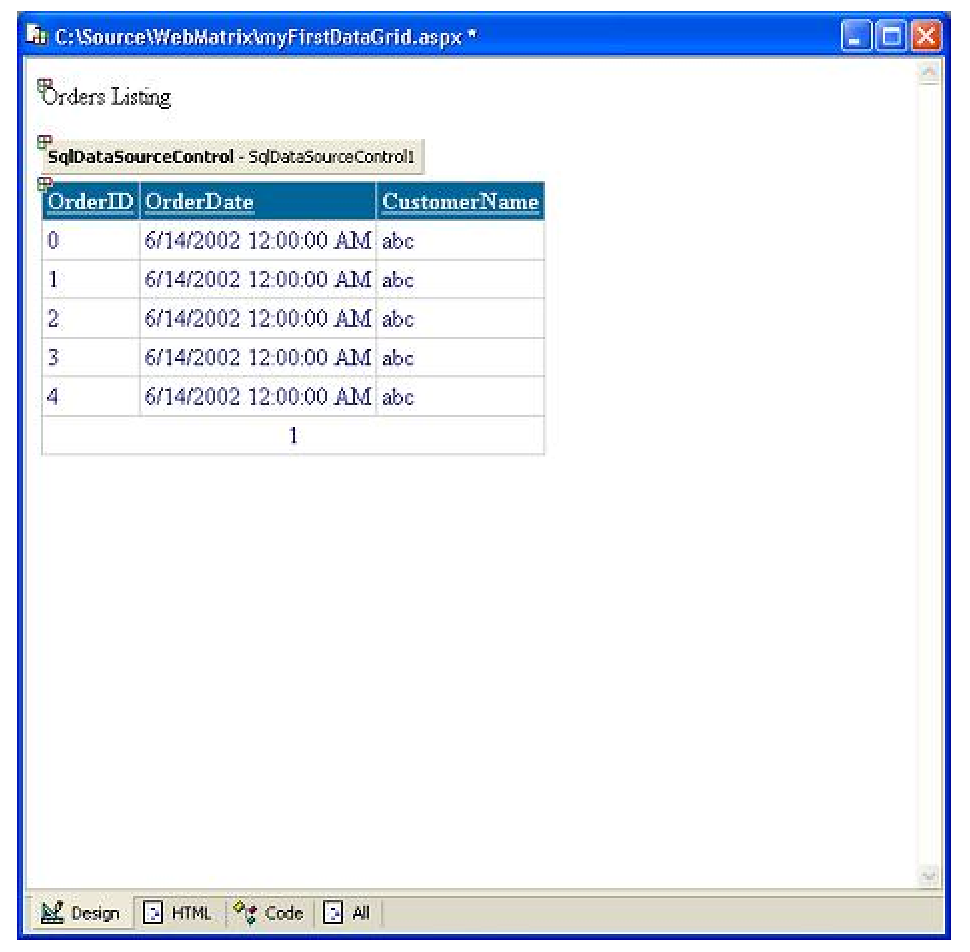


الان نريد أن نعرض البيانات التى تم حفظها فى قاعدة البيانات Orders داخل صفحة Aspx نقوم بالتالى :

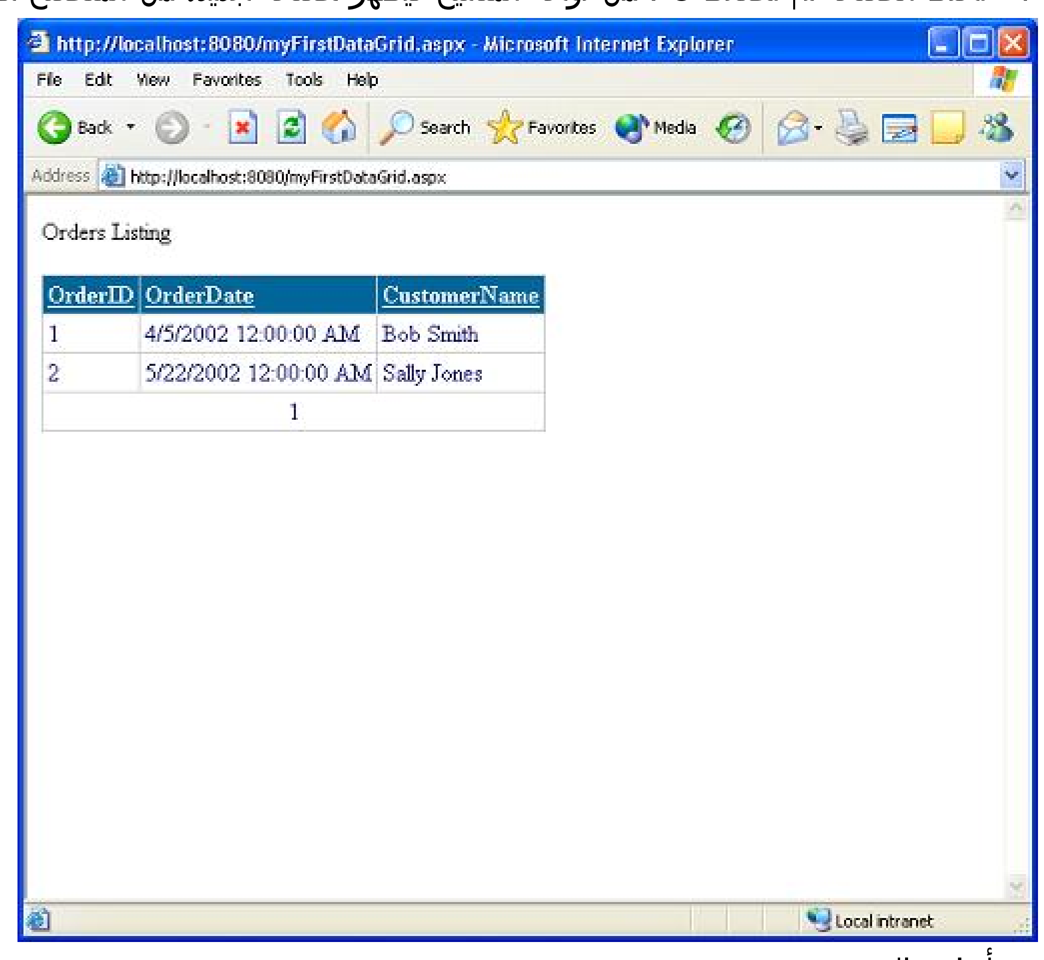
١- اعمل ملف جديد و ليكن اسمة myFirstDataGrid.aspx ستفتح الصفحة فى وضع التصميم اكتب Orders Listing ثم اضغط Enter ثم انقر على تبويب Data فى نافذة Workspace ثم انقر على New Connection من شريط الادوات فى النافذة يظهر مربع حوار Connect to Database اختر من القائمة المنسدلة قاعدةالبيانات Orders ثم انقر على ok .



٢- اختر الجدول Orders ثم اسحبة داخلة الصفحة فيظهر على الصفحة SQLDataSourceControl و MXDataGrid يقوم البرنامج بعمل اللازم لملئ
ال MXDataGrid بالبيانات الموجودة فى الجدول بالاعتماد على المعلومات الموجودة لدية عن الاتصال بقاعدة البيانات .



٣- احفظ الصفحة ثم اضغط F5 من لوحة المفاتيح فيظهر صفحة جديدة من المتصفح كالتالى

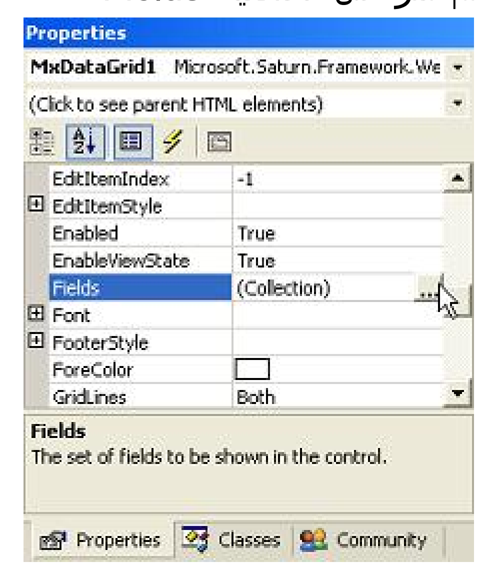


قم بأغلاق الصفحة .

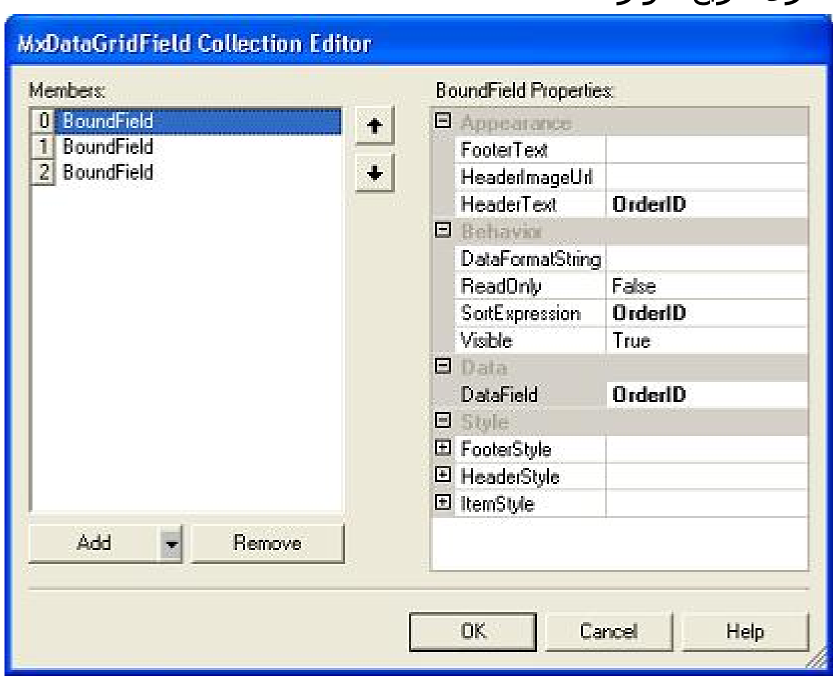
٥- تنسيق ال MxDataGrid يكون بالنقر على MxDataGrid ثم النقر على الخاصية AllowPaging فى نافذة الخصائص و اختر False



ثم انقر على الخاصية Fields

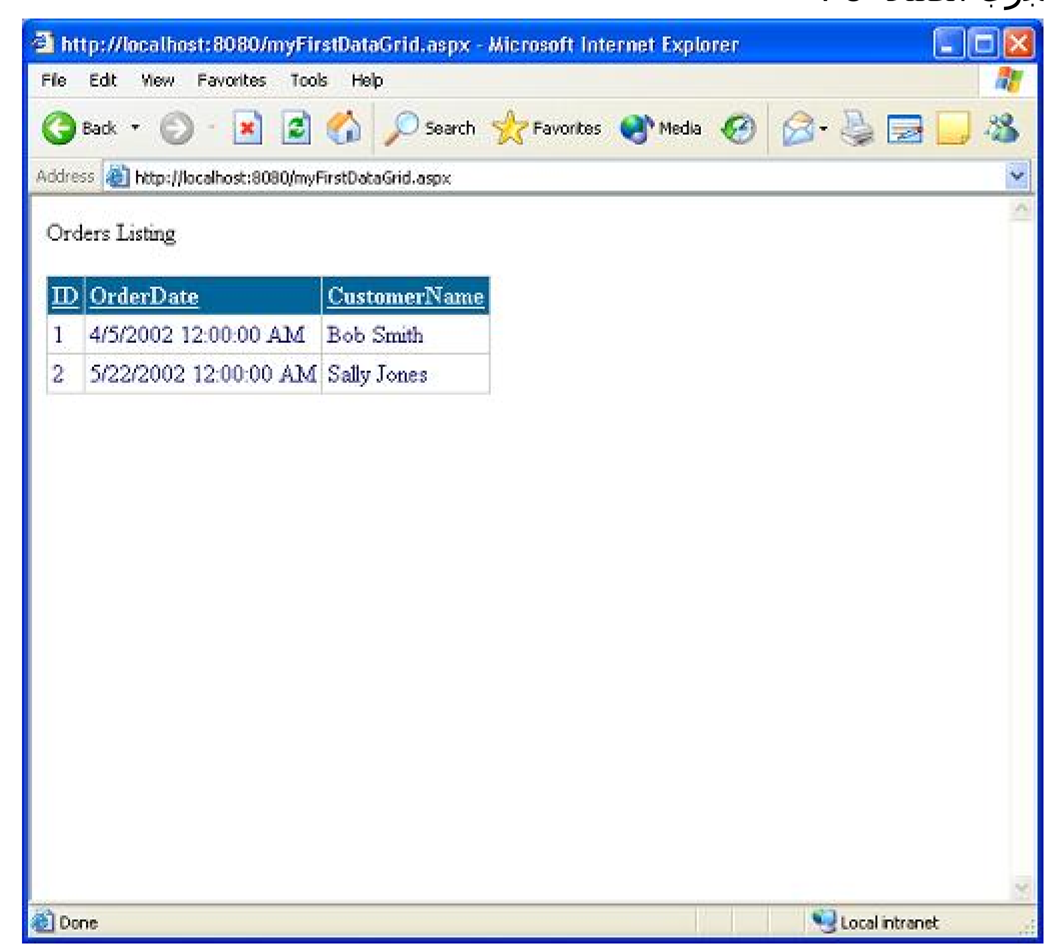


سترى مربع حوار Collection Editor



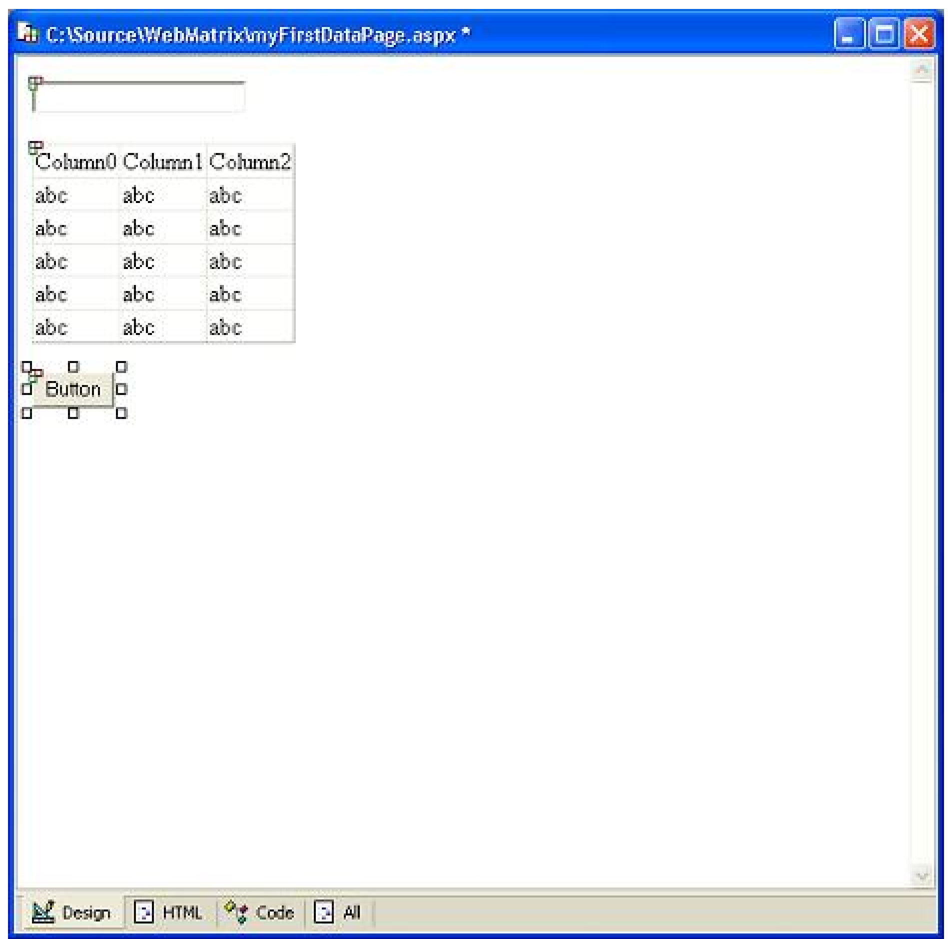
غير الخاصية HeaderText بالنسبة ل OrderID الى ID ثم اضغط Ok.

جرب الصفحة F5



- <u>استخدام Using Data Code Builder</u>

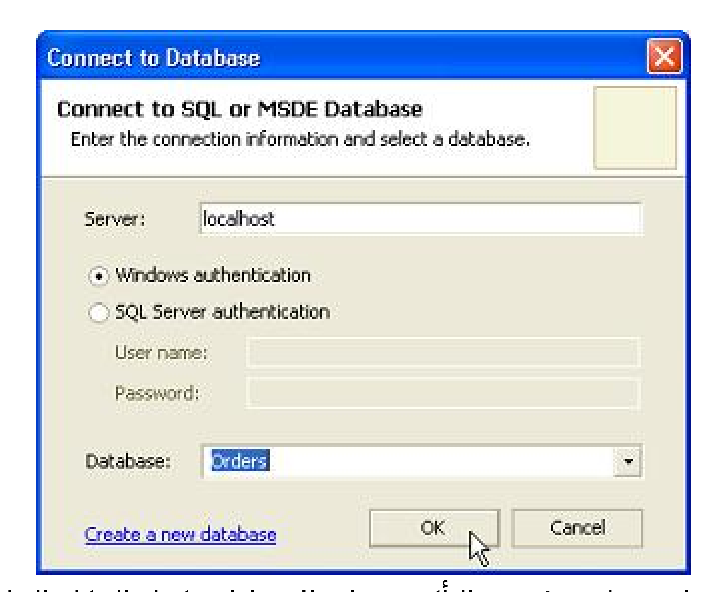
١- قم بعمل ملف جديد اسمة myFirstDataPage.aspx فى وضع التصميم اسحب a TextBox control, a DataGrid control, and a Button control و اتركهم على الصفحة و افصل بينهم ب Enter .



٢- انقر على تبويب Code و انقر على تبويب Code Builders من صندوق الادوات Toolbox ثم اسحب SELECT Data Method و اتركة فى الصفحة انظر الشكل

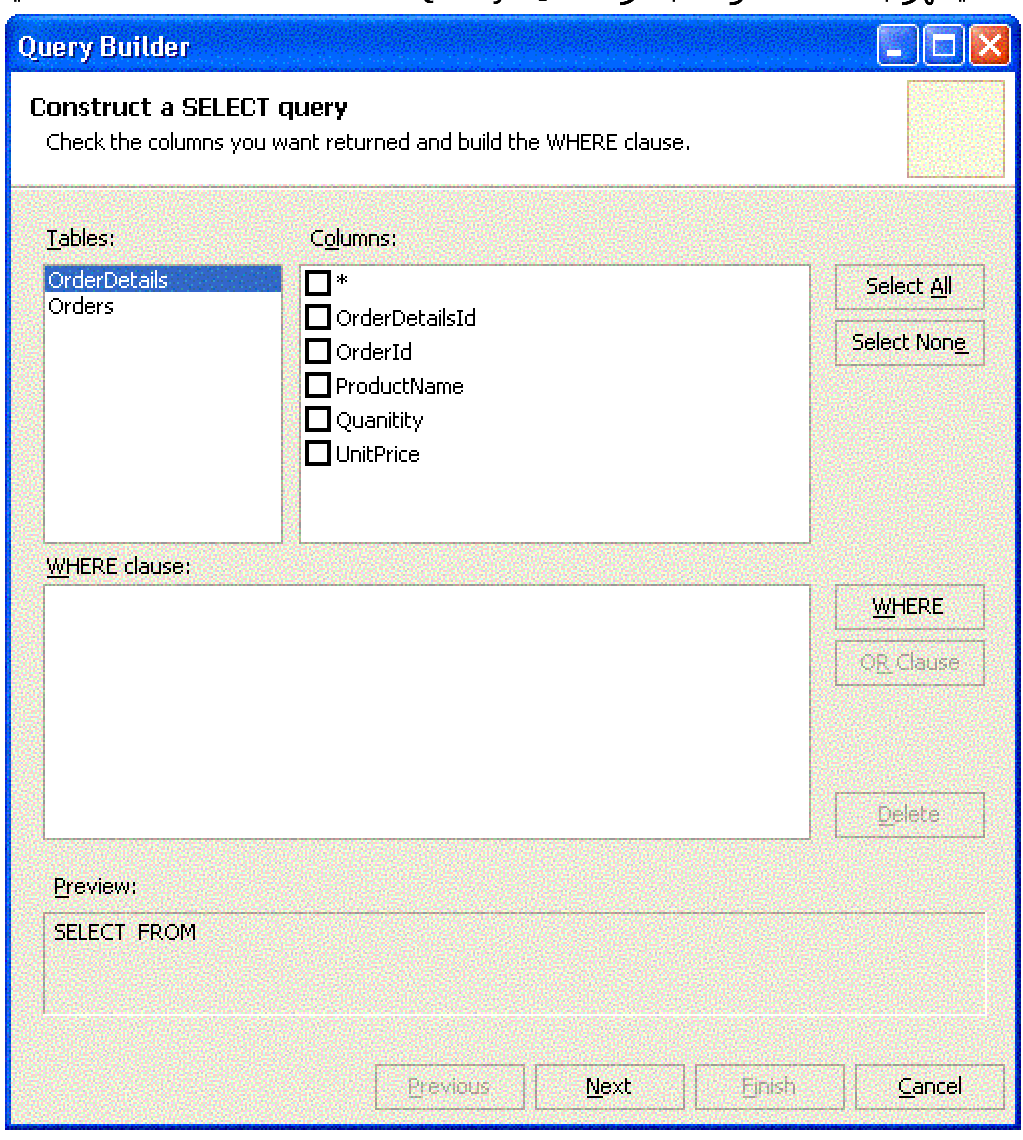


٣- يظهر مربع حوار Connect to Database

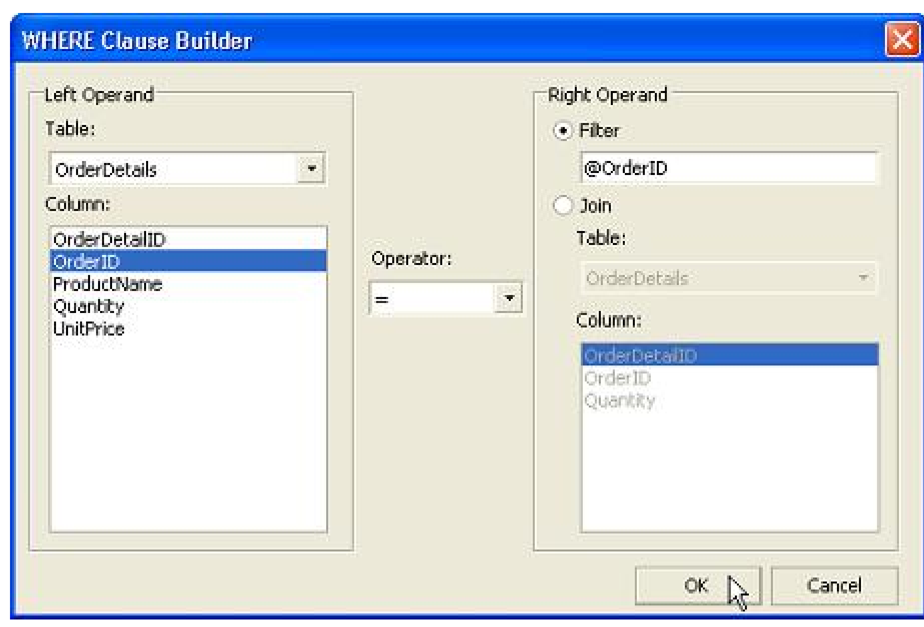


انقر على okبعد التأكد من ان الاعدادات كما بالشكل السابق .

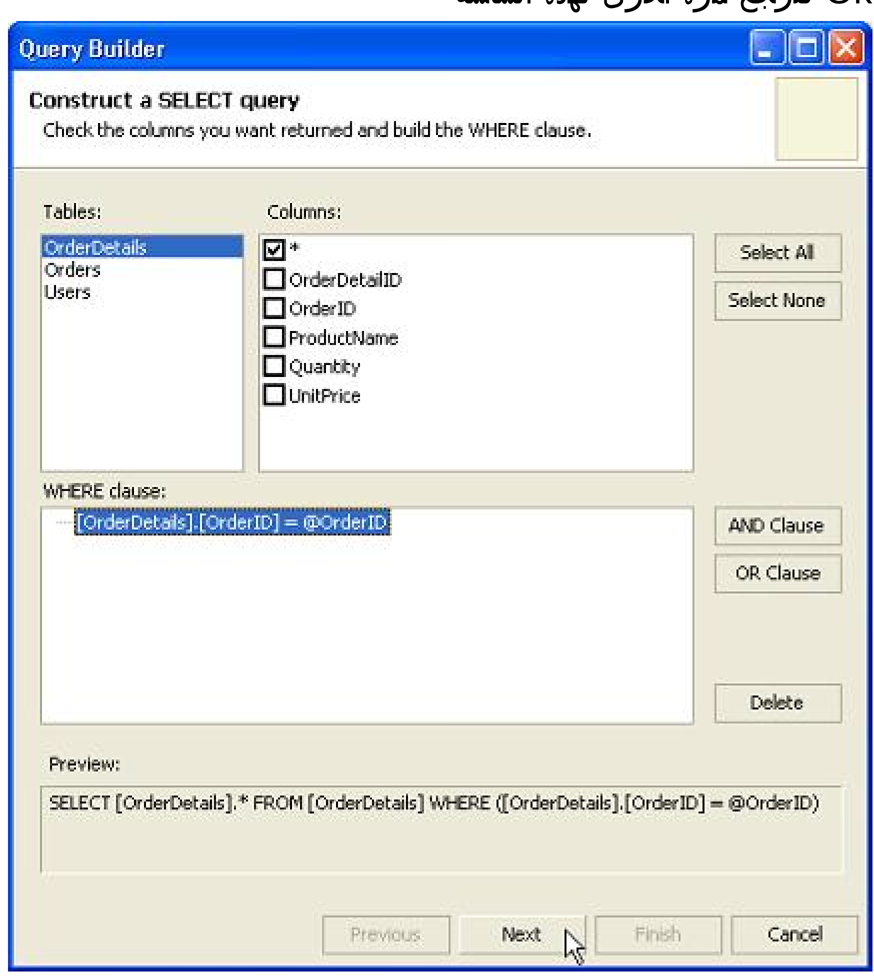
٤- يظهر بعد ذلك خطوات جاهزة تسمى Construct SELECT Query كما يلى



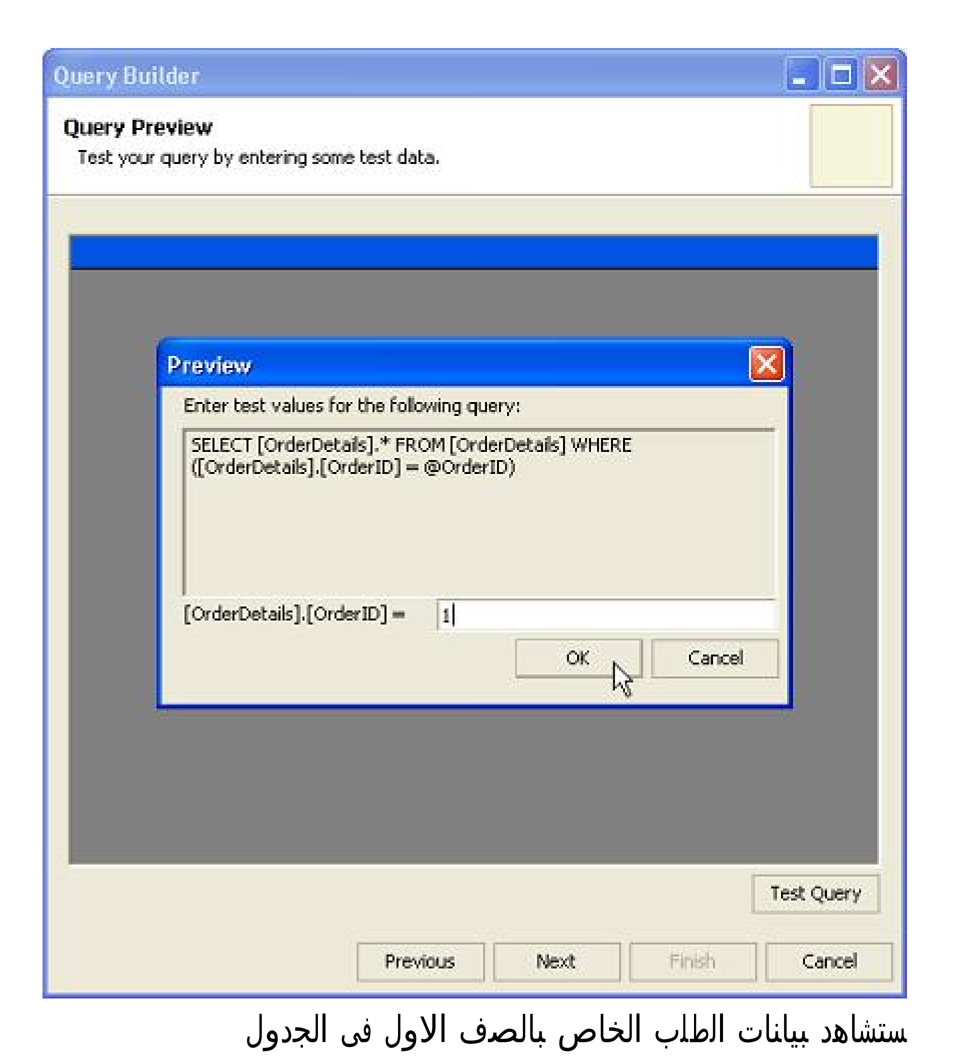
انقر على OrderDetails فى الجزء الخاص ب Tables ثم اختر علامة النجمة * و معناها هنا اختر اظهار كل الاعمدة فى الجدول ثم انقر على WHERE



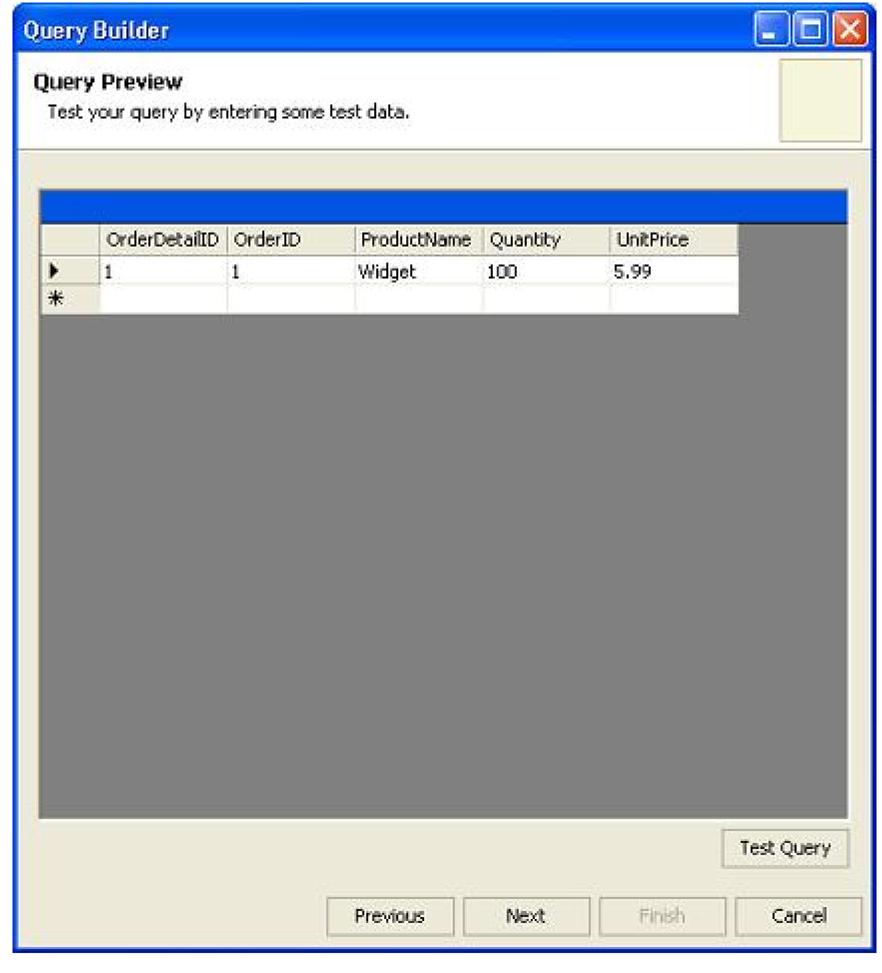
اختر OrderID من Column ثم انقر على Filter و اكتب فى الخانة اسفلة @OrderID ثم انقر على Ok فترجع مرة اخرى لهذة الشاشة



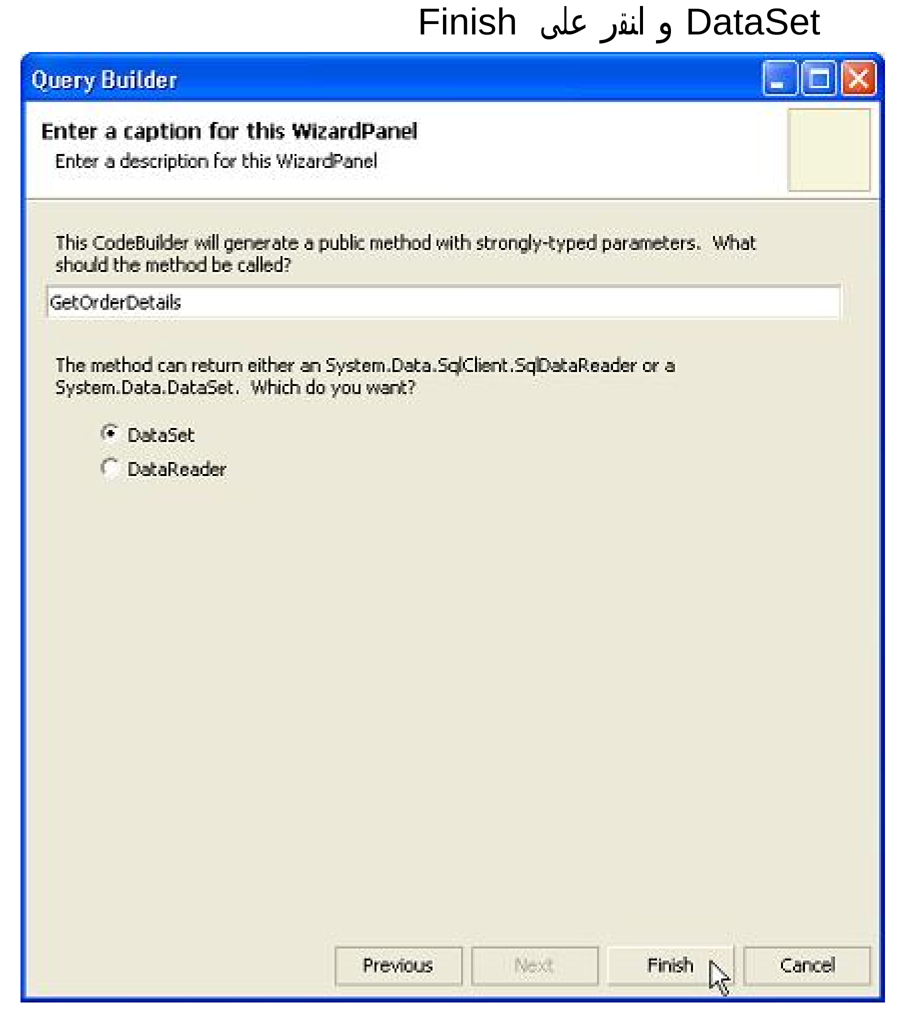
٥- انقر على Next تظهر الخطوة الثانية و هى المعاينة Query Preview انقر على Test Query لتجرب جملة الاستعلام السابقة و اكتب ١ فى OrderID ثم ok



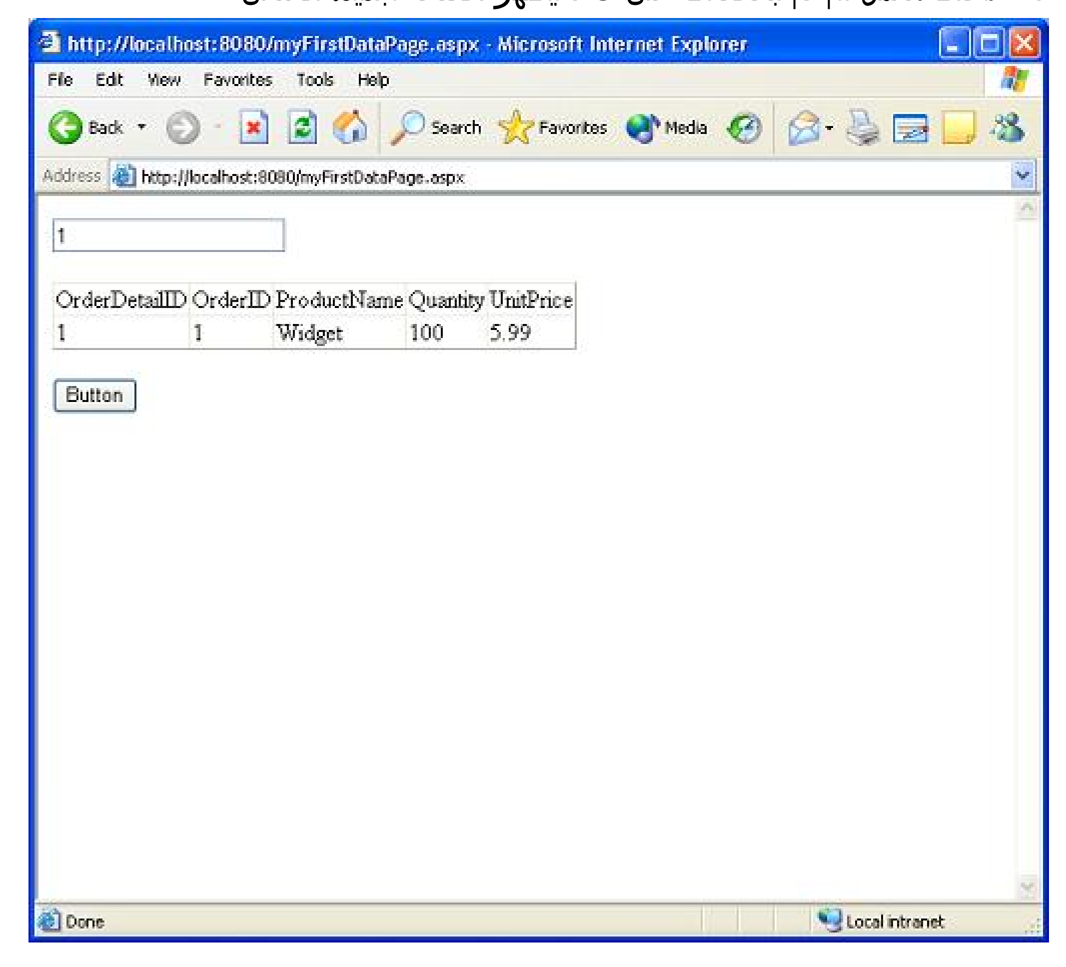
ستشاهد بيانات الطلب الخاص بالصف الاول فى الجدول



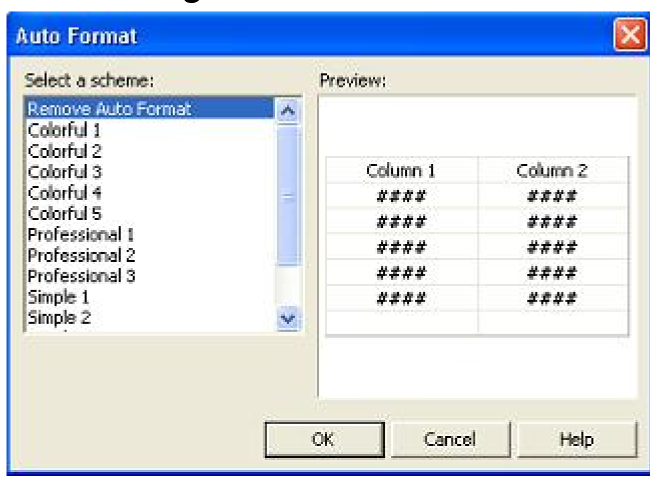
٦- انقر على Next فتظهر الخطوة الثالثة فاكتب GetOrderDetails فى خانة method name ثم اختر DataSet و انقر على Finish



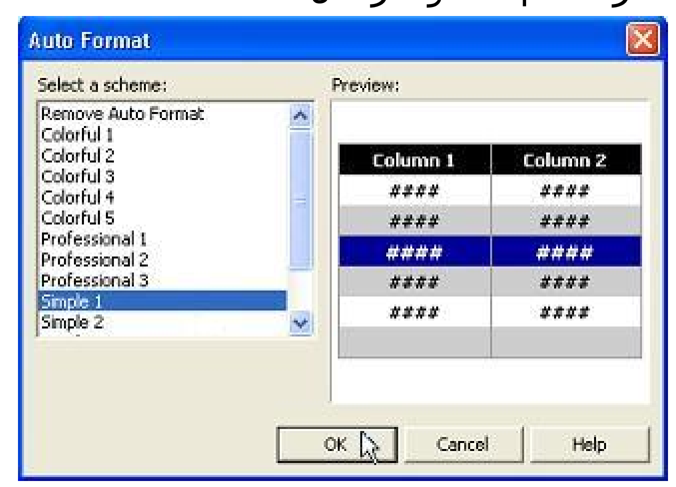
٧- احفظ العمل ثم قم بالضغط على F5 يظهر صفحة جديدة كالتالى



٨- اغلاق الصفحة و ارجع لوضع التصميم و قم بتنسيق ال DataGrid بالضغط على Auto Format من نافذة الخصائص



اختر Simple 1 و انقر على OK



<u>عرض البيانات بواسطة لغة Xml من خلال المتصفح :</u>

```
<% @Page ContentType='text/xml' language='c#' debug='true' %>
<% @Import namespace='System.Data' %>
<% @Import namespace='System.Data.SqlClient' %>

<SCRIPT runat='server'>
void Page_Load(Object Sender,EventArgs e)
{
SqlConnection cn;
SqlDataAdapter da;
DataSet ds;
String strSQL;

strSQL = "SELECT TOP 10 au_fname, au_lname FROM authors";
cn = new
SqlConnection("server=localhost;uid=sa;pwd=;database=pubs;");
da = new SqlDataAdapter(strSQL, cn);
ds = new DataSet();
da.Fill(ds, "Authors");
```

```
Response.Write(ds.GetXml());
}
</SCRIPT>
```

تحميل ملف xml داخل صفحة الويب باستخدام XmlDocument's .Load()

ملف الxml اسمة books.xml

```
<BOOKS>
<BOOK>
<TITLE>C# and Asp.net </TITLE>
<AUTHOR id='101' location='egypt'>michael nabil</AUTHOR>
<AUTHOR id='107' location='cairo'>frank nabil</AUTHOR>
</BOOK>
</BOOKS>
```

الصفحة التى سنعرض فيها محتويات الملفbooks.xml

```
<% @Page language="C#" debug="true" %>
<%@ Import Namespace="System.Xml" %>

<SCRIPT runat='server'>
void Page_Load(Object Sender,EventArgs e)
{
  XmlDocument xd = new XmlDocument();
  xd.Load(Server.MapPath("books.xml"));
  Response.Write (xd.OuterXml);
  xd = null;
}
</SCRIPT>
```

تحميل ملف الxml موضوع على السيرفر

```
<% @Page language="C#" debug="true" %>
<%@ Import Namespace="System.Xml" %>
<SCRIPT runat='server'>
void Page_Load(Object Sender,EventArgs e)
{
  XmlDocument xd = new XmlDocument();
  xd.Load("http://www.myserver.com/books.xml");
  Response.Write (xd.OuterXml);
  xd = null;
}
</SCRIPT>
```

و يمكن استخدام ChildNodes

```
<% @Page language="C#" debug="true" %>
<%@ Import Namespace="System.Xml" %>

<SCRIPT runat='server'>
void Page_Load(Object Sender,EventArgs e)
{
XmlDocument xd = new XmlDocument();
```

```
XmlNode ndBook;

xd.Load(Server.MapPath("books.xml"));
ndBook = xd.FirstChild["BOOK"];

foreach(XmlNode nd in ndBook.ChildNodes)
{
if(nd.Name == "AUTHOR")
Response.Write("The author's name is " + nd.InnerText + "<BR>");
}
}
</SCRIPT>
```

استخدام XmlTextReader

```
<% @Page language="C#" debug="true" %>
<%@ Import Namespace="System.Xml" %>
<SCRIPT runat='server'>

void Page_Load(Object Sender,EventArgs e)
{
   XmlTextReader xr = new XmlTextReader(Server.MapPath("books.xml"));
   Boolean bTitle = false;

   while(xr.Read())
   {
     switch(xr.NodeType)
     {
       case XmlNodeType.Element:
         if(xr.Name == "TITLE")
           bTitle = true;
         break;
       case XmlNodeType.Text:
         if(bTitle)
         {
           Response.Write("Book title: " + xr.ReadString());
           bTitle = false;
         }
         break;
     }
   }
}
</SCRIPT>
```

الكتابة داخل ملف xml باستخدام XmlTextWriter Object

```
<%@ Import Namespace="System.Xml" %>
<SCRIPT runat='server'>

void Page_Load(Object Sender,EventArgs e)
{
XmlTextWriter xw = new XmlTextWriter(Server.MapPath("books2.xml"),
Encoding.UTF8);
try
{
xw.WriteStartDocument();
xw.WriteStartElement("BOOK");
```

```
xw.WriteElementString("TITLE", "asp.net");
xw.WriteEndDocument();
Response.Write("Your file has been written.");
}
catch(Exception ex)
{
Response.Write("Exception: " + ex.Message);
}
finally
{
xw.Flush();
xw.Close();
}
}
</SCRIPT>
```

تصفح ملف ال xml باستخدام XmlNodeReader Object

```
<%@ Import Namespace="System.Xml" %>
<SCRIPT runat='server'>

void Page_Load(Object Sender,EventArgs e)
{
  XmlDocument xd = new XmlDocument();
  xd.Load(Server.MapPath("books.xml"));
  XmlNodeReader xn = new XmlNodeReader(xd);

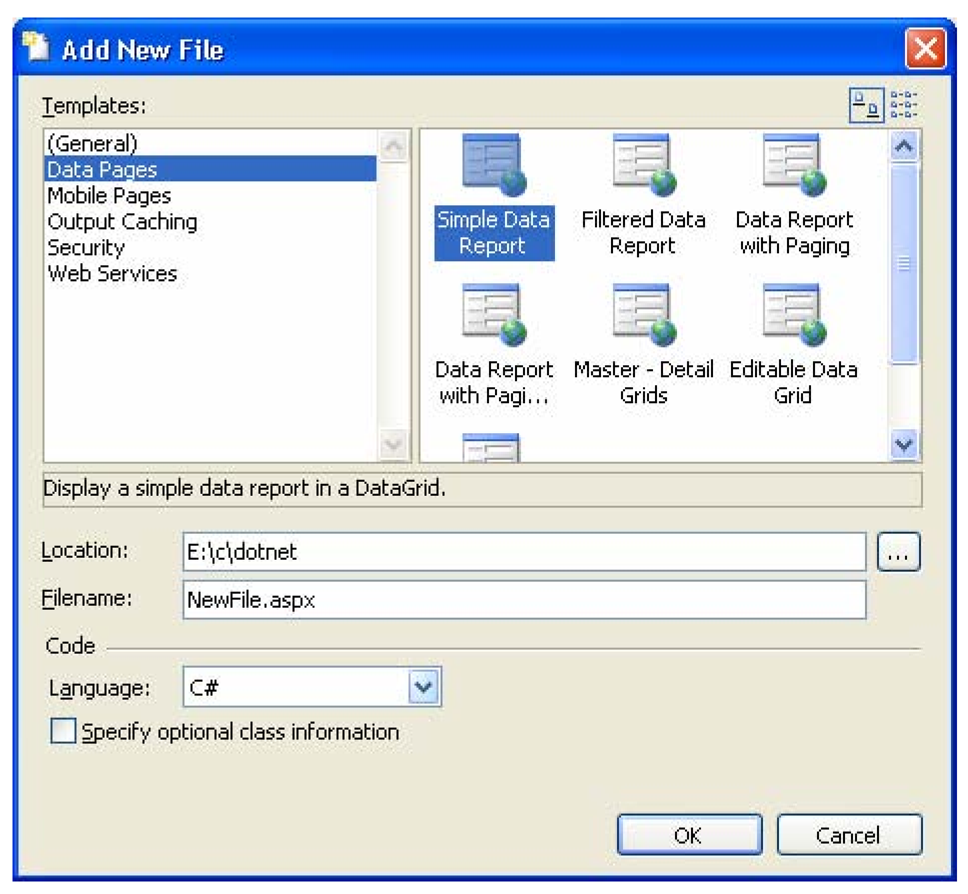
  while(xn.Read())
  {
    Response.Write(xn.Name + " - " + xn.Value + "<BR>");
  }

}
</SCRIPT>
```

هناك تسأل الان الا يمكن ان اعمل صفحات جاهزة للأدخال البيانات و تقديم تقارير لزوار الموقع بطريقة الية من خلال البرنامج ؟

نعم يمكننا هذا و الطريقة سهلة جدا اتبع الاتى :

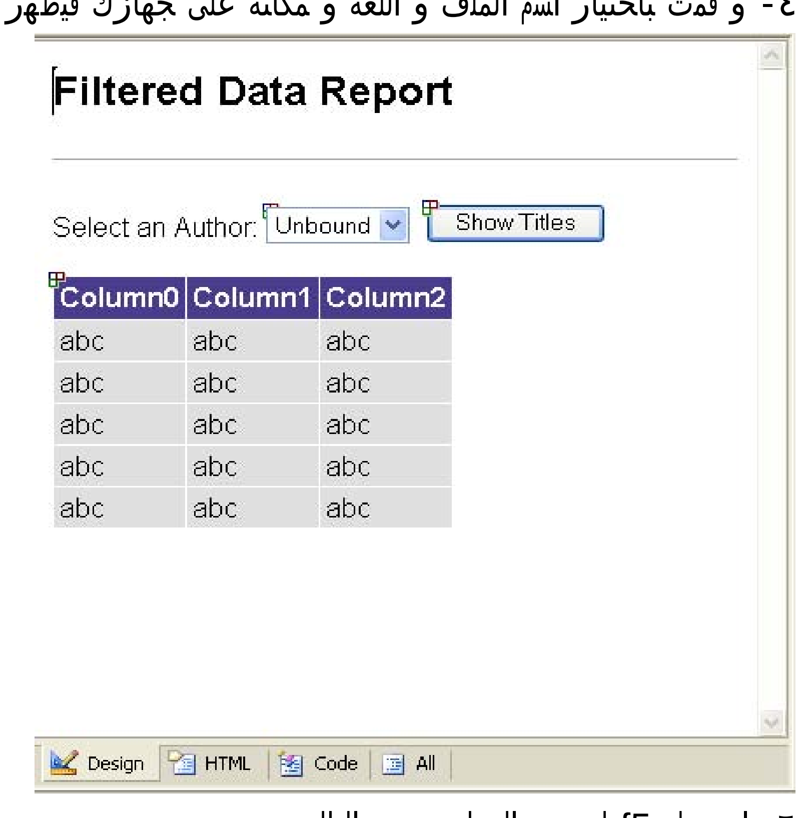
١ - انقر على ايقونة ملف جديد فتظهر لك الشاشة التالية



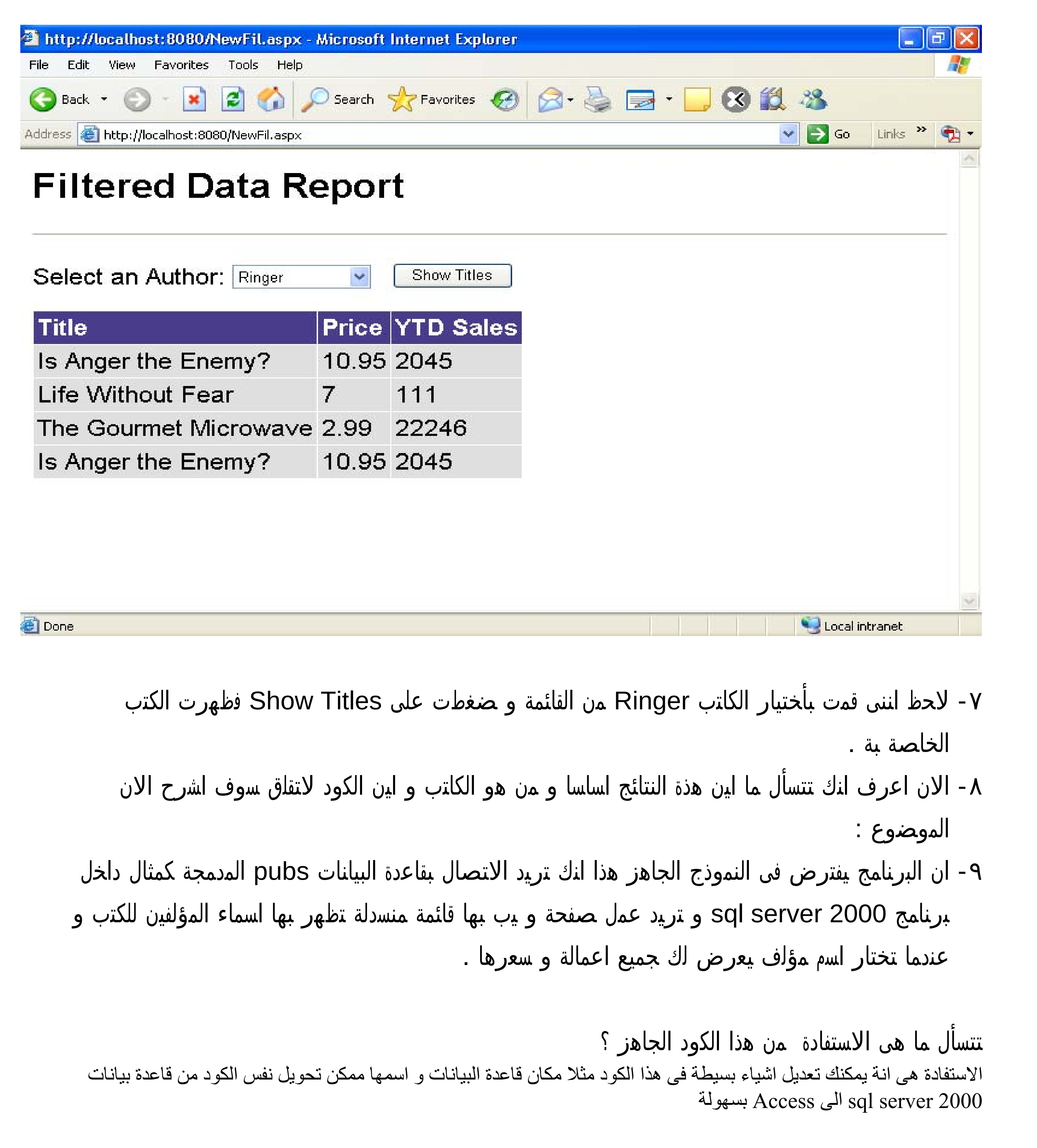
٢- اختر Data pages من تبويب Templates

٣- امامك اختيارات كثيرة لصفحات جاهزة مثلا اذا اخترنا Filterd Data Report

٤- و قمت باختيار اسم الملف و اللغة و مكانة على جهازك فيظهر لك الصفحة التالية :



٦- اضغط f5 لتجربة العمل سترى التالى:



٧- لاحظ اننى قمت بأختيار الكاتب Ringer من القائمة و ضغطت على Show Titles فظهرت الكتب الخاصة بة .

٨- الان اعرف انك تتسأل ما اين هذة النتائج اساسا و من هو الكاتب و اين الكود لاتقلق سوف اشرح الان الموضوع :

٩- ان البرنامج يفترض فى النموذج الجاهز هذا انك تريد الاتصال بقاعدة البيانات pubs المدمجة كمثال داخل برنامج sql server 2000 و تريد عمل صفحة و يب بها قائمة منسدلة تظهر بها اسماء المؤلفين للكتب و عندما تختار اسم مؤلف يعرض لك جميع اعمالة و سعرها .

تتسأل ما هى الاستفادة من هذا الكود الجاهز ؟

الاستفادة هى انة يمكنك تعديل اشياء بسيطة فى هذا الكود مثلا مكان قاعدة البيانات و اسمها ممكن تحويل نفس الكود من قاعدة بيانات sql server 2000 الى Access بسهولة

<u>WebControls وربطها بالبيانات</u>

سأشرح أداوت ال Web Controls المتقدمة عما سبق و التي ترتبط بمصادر للبيانات كقواعد البيانات أو ملفات XML موجودة على السيرفر أو على أماكن ما على الشبكة.

الأدوات تختلف و لكن طريقة الربط متشابهة جدا و الان مع الاداة الاولى DropDownList

```
<asp:dropdownlist id="dl" runat="server">
<asp:listitem text="image1" value="images/image1.gif" />
<asp:listitem text="image2" value="images/image2.gif" />
<asp:listitem text="image3" value="images/image3.gif" />
</asp:dropdownlist>
```

هذه كانت الطريقة العادية و لكن ساكتب مثالا يوصلنا بجدول للبيانات ثم نعرضها في القائمة المنسدلة:

```
<%@ Page Language="C#" Debug="True" %>
<%@ import Namespace="System.Data.OleDb" %>
<%@ import Namespace="System.Web.UI" %>
<%@ import Namespace="System.Web.UI.WebControls" %>
<script runat="server">

void Page_Load()
{

object rcv=Request.QueryString["DL"];
if(rcv!=null) {
Response.Write(rcv + "<BR>");
Response.End();
}
else
{
string prov="Provider=Microsoft.Jet.OleDb.4.0;Data Source=" + Server.MapPath("db.mdb");
string SQL="SELECT * FROM Categories";
OleDbConnection cn = new OleDbConnection(prov);
OleDbCommand cmd = new OleDbCommand(SQL,cn);


cn.Open();


OleDbDataReader reader = cmd.ExecuteReader();

DL.DataSource= reader;
DL.DataBind();

cn.Close();
}
}

</script>
<html>
<head>
</head>
<body>
<form action="repeater.aspx" method="get" runat="server">
<asp: DropDownList id="DL" runat="server" datavaluefield="id" datatextfield="category"></asp:
DropDownList>
</form>
</body>
</html>
```

فكرة المثال هي عمل نموذج Form يرسل البيانات عبر المتصفح و يحوي على قائمة منسدلة بها التصنيفات الموجودة في قاعدة البيانات Categories و زر يقوم بإتمام العملية حيث ننتقل لنفس الصفحة ولكن لا يعرض إلا رقم التصنيف الذي إخترناه.

بداية دعونا ننظر للجزء الذي كتبناه من أجل إدراج الأداة:

```
<asp: DropDownList id="DL" runat="server" datavaluefield="id" datatextfield="category"></asp:
DropDownList>
```

كما نرى أن الإختلاف واضح حيث لم نستخدم أيا من ال ListItem كما سبق و قمنا بتحديد خاصيتين الأولى هي datavaluefield و هي تأخذ إسم العمود في جدول البيانات و هو كان رقم التصنيف id و الخاصية الثانية هي datatextfield و هي ستعرض النص أمامنا على الشاشة حسب حقل category الموجود في جدول البيانات.

الأكواد البرمجية نعرفها جميعا و هي عن كيفية عمل DataReader و جعله يخزن مجموعة بيانات و لكن الجديد هو ربط القائمة المنسدلة التي أسميتها DL من خلال الخاصية id و هذه هي الأكواد الجديدة:

```
DL.DataSource= reader;
DL.DataBind();
```

حيث حددت الخاصية DataSource و هي مصدر البيانات على أنها محتويات ال Reader ثم قمت بربط البيانات فعليا مع الأداة من خلال الطريقة. DataBind

أعتقد أن الطريقة سهلة و يبقى أنني في أعلى الكود عرفت متغير من نوع Object بإسم rcv

```
object rcv=Request.QueryString["DL"];
```

يقوم بإستقبال القيمة التي أرسلت للمتصفح و الخاصة برقم التصنيف و طبعا النموج كان معدا عند خاصية ال Method على الطريقة Getحيث تمرر البيانات إلى شريط العنوان في المتصفح ثم قمت بجملة شرطية تفيد في فحص المتغير إن كان له قيمة أي تم إختيار قسم من القائمة و ضغط الزر فيقوم بكتابتها و إيقاف تنفيذ باقي الصفحة كي لا تظهر عن طريق الطريقة End الخاصة بالكائن Responseو إلا فسوف يدخل لجملة الإستثناء و التي تقوم بجلب البيانات و إسنادها إلى القائمة و إظهارها للمستخدم و للعلم إن لم يتم الربط Data Binding فلن تظهر أية بيانات في القائمة المنسدلة و هذا شيء طبيعي.

RadioButtonList

الطريقة مشابهة لما سبق و سأكتفي بذكر مثال فقط و بنفس ما شرحت المثال السابق سيكون هنا

```
<%@ Page Language="C#" Debug="True" %>
<%@ import Namespace="System.Data.OleDb" %>
<%@ import Namespace="System.Web.UI" %>
<%@ import Namespace="System.Web.UI.WebControls" %>
<script runat="server">

void Page_Load()
{


string prov="Provider=Microsoft.Jet.OleDb.4.0;Data Source=" + Server.MapPath("db.mdb");
string SQL="SELECT * FROM Categories";
OleDbConnection cn = new OleDbConnection(prov);
OleDbCommand cmd = new OleDbCommand(SQL,cn);


cn.Open();


OleDbDataReader reader = cmd.ExecuteReader();

rd.DataSource= reader;
rd.DataBind();

cn.Close();

}

</script>
<html>
<head>
</head>
<body>
<form runat="server">
<asp:RadioButtonList id="rd" runat="server" datavaluefield="id"
datatextfield="category"></asp:RadioButtonList>
</form>
</body>
</html>
```

فقط إنتبه للخاصيتين datavaluefield و الخاصية. datatextfield

CheckBoxList

أيضا سأكتفي بطرح مثال توضيحي

```
<%@ Page Language="C#" Debug="True" %>
<%@ import Namespace="System.Data.OleDb" %>
<%@ import Namespace="System.Web.UI" %>
<%@ import Namespace="System.Web.UI.WebControls" %>
```

```
<script runat="server">

void Page_Load()
{


string prov="Provider=Microsoft.Jet.OleDb.4.0;Data Source=" + Server.MapPath("db.mdb");
string SQL="SELECT * FROM Categories";
OleDbConnection cn = new OleDbConnection(prov);
OleDbCommand cmd = new OleDbCommand(SQL,cn);


cn.Open();


OleDbDataReader reader = cmd.ExecuteReader();

cb.DataSource= reader;
cb.DataBind();



cn.Close();

}
</script>
<html>
<head>
</head>
<body>
<form runat="server">
<asp:CheckBoxList id="cb" runat="server" datavaluefield="id"
datatextfield="category"></asp:CheckBoxList>
</form>
</body>
</html>
```

Repeater

بصراحة هذه الأداة مهمة جدا في تطبيقاتنا و هي سهلة الإستخدام و يتم ربطها كما سبق و سأذكر هنا مثلا يقوم بعرض الأقسام الموجودة في جدول التصنيفات Categories و سأجعل كل نتيجة تظهر على شكل وصلة تذهب بنا لصفحة تعرض البيانات الخاصة بالتصنيف حيث نمرر القيمة الخاصة بال Category ID لتلك الصفحة.

دعونا نرى المثال:

```
<%@ Page Language="C#" Debug="True" %>
<%@ import Namespace="System.Data.OleDb" %>
<%@ import Namespace="System.Web.UI" %>
<%@ import Namespace="System.Web.UI.WebControls" %>
<script runat="server">

void Page_Load()
{


string prov="Provider=Microsoft.Jet.OleDb.4.0;Data Source=" + Server.MapPath("db.mdb");
string SQL="SELECT * FROM Categories";
OleDbConnection cn = new OleDbConnection(prov);
OleDbCommand cmd = new OleDbCommand(SQL,cn);
```

```
cn.Open();

OleDbDataReader reader = cmd.ExecuteReader();

rp.DataSource= reader;
rp.DataBind();

cn.Close();

}

</script>
<html>
<head>
</head>
<body>
<form runat="server">
<asp:Repeater id="rp" runat="server">
<Itemtemplate>
<a href="more.aspx?id=<%# DataBinder.Eval(Container.DataItem,"id")%>"> <%#
DataBinder.Eval(Container.DataItem,"category")%> <a />
<br />
</Itemtemplate>
</asp:Repeater>
</form>
</body>
</html>
```

الكود واضح جدا بالأعلى و هو تقليدي .. دعونا نرى منطقة ال HTML

قمنا بإدراج أداة Repeater كالتالي:

```
<asp:Repeater id="rp" runat="server">
</asp:Repeater>
```

و من أجل إظهار محتويات ال Repeater يجب إضافة الوسوم ItemTemplate حيث سنقوم بداخلها بكتابة ما نريد ليصبح الكود كالتالي:

```
<asp:Repeater id="rp" runat="server">
<Itemtemplate>
//نكتب هنا ما نريد
</Itemtemplate>
</asp:Repeater>
```

والآن كيف سنعرض الأقسام ؟؟ سنضيف التالي:

```
<asp:Repeater id="rp" runat="server">
<Itemtemplate>
<%# DataBinder.Eval(Container.DataItem,"category")%>
<br>
</Itemtemplate>
</asp:Repeater>
```

سيظهر كل قسم في سطر و لكن من أجل جعله وصلة يجب إضافة الوسم <a> أو نستخدم الأداة hyperlink كما أخذنا في دروس ال webControls ليصبح الكود كما يلي:

```
<asp:Repeater id="rp" runat="server">
<Itemtemplate>
<a href="more.aspx?id=<%#DataBinder.Eval(Container.DataItem,"id")%>">
<%# DataBinder.Eval(Container.DataItem,"category")%>
<a />
<br />
</Itemtemplate>
</asp:Repeater>
```

أعتقد أن الفكرة سهلة للغاية.

أداة ال DataList من الأدوات المهمة جدا في تطبيقات ال ASP.NET و التي تقوم بعرض البيانات للزائر بعدة أشكال ممكنة من خلال إمكانتها المتاحة سواء في التصميم أو في دعمها للأحداث و إحتواء أدوات أخرى. في البداية ال DataList تدعم إستخدام القوالب أو ما يعرف في الويب بال Templatesو تقريبا تشبه ما ذكرناه في موضوع ال Repeater و لكن بالأصح هي أقرب لما تدعمه أداة ال DataGrid حيث تملك نفس القوالب.

القوالب الموجودة داخل ال: DataList

1.HeaderTemplate:

تحدد كيف سيظهر رأس ال DataList أي المنطقة العلوية للعرض.

2.ItemTemplate:

و هي المنطقة الأكثر حيوية حيث يتم تنسيق البيانات التي ستعرض فيها.

3.AlternateItemTemplate:

يتم فيها تنسيق العناصر التي تظهر مرة بعد مرة خلال عرض سجلات البيانات.

4.SeparatorTemplate:

و هي المنطقة الفاصلة بين السجلات المعروضة و قلنا بالعادة نستخدم فيها الوسم.

5.FooterTemplate:

و فيها يتم تنسيق ذيل ال Data List و إعداده إن استلزم الأمر.

6.SelectedItemTemplate:

تتحكم في تنسيق العنصر الذي يتم إختياره.

7.EditItemTemplate:

تقوم بتنسيق العنصر الذي تم إختياره لإجراء التعديل عليه بشكل معين.

ربط ال DataList بالبيانات:

نفس طريقة ال DataBinding التي ذكرناها في ال Repeater تستخدم هنا سواء إستخرجنا البيانات عن طريق DataReader أو DataSet و هذا مثال يوضح ذلك:

CODE

```
<%@ Page Language="C#" Debug="True" %>
<%@ import Namespace="System.Data.OleDb" %>
<script runat="server">
void Page_Load()
{

string prov="Provider=Microsoft.Jet.OleDb.4.0;Data Source=" + Server.MapPath("db.mdb");
string SQL="SELECT * FROM Categories";
OleDbConnection cn = new OleDbConnection(prov);
OleDbCommand cmd = new OleDbCommand(SQL,cn);

cn.Open();

OleDbDataReader reader = cmd.ExecuteReader();

dl.DataSource= reader;
dl.DataBind();

cn.Close();

}

</script>
<html>
<head>
</head>
<body>
<form runat="server">
<asp: DataList id="dl" runat="server">
<ItemTemplate>
<%# DataBinder.Eval(Container.DataItem,"category")%>
</ItemTemplate>
</asp: DataList>
</form>
</body>
</html>
```

لننظر بداية إلى جزء ال HTML حيث أدرجنا أداة ال: DataList

CODE

```
<asp: DataList id="dl" runat="server">
</asp: DataList>
```

ثم ذهبنا للحدث Page_Load و أضفنا أكواد من أجل الإتصال بقاعدة البيانات و إستخراج السجلات من جدول Categories عن طريق الأداة. DataReader

بعد ذلك قمنا بربط الأداة DataList و التي أسميتها dl بالبيانات من خلال التالي:

CODE

```
dl.DataSource= reader;
dl.DataBind();
```

و من أجل عرض أسماء التصنيفات من الجدول نحتاج لكتابة التالي:

CODE

```
<%# DataBinder.Eval(Container.DataItem,"category")%>
```

و طبعا يجب أن يكون داخل ال ItemTemplate لنصحل على النتيجة المطلوبة و هي إظهار جميع أسماء التصنيفات الموجودة داخل الجدول.Categories

لنفرض أننا نود تغيير ترتيب الظهور و إستخدام دالة نمرر لها قيم معينة و نضيف وصلة تشعبية HyperLink فماذا نفعل ؟

دعونا نرى هذا المثال:

CODE

```
<%@ Page Language="C#" Debug="True" %>
<%@ import Namespace="System.Data.OleDb" %>
<script runat="server">

public object IsActive(object vl)
{
string ac="";
if(vl.ToString()=="True")
ac="Yes";
else
ac="No";
return ac;
}


void Page_Load()
{
```

```
string prov="Provider=Microsoft.Jet.OleDb.4.0;Data Source=" +
Server.MapPath("db.mdb");
string SQL="SELECT * FROM Categories";
OleDbConnection cn = new OleDbConnection(prov);
OleDbCommand cmd = new OleDbCommand(SQL,cn);


cn.Open();


OleDbDataReader reader = cmd.ExecuteReader();

dl.DataSource= reader;
dl.DataBind();

cn.Close();

}

</script>
<html>
<head>
</head>
<body>
<form runat="server">
<asp: DataList id="dl" runat="server" RepeatDirection="Horizontal"
repeaterlayou="table" borderwidth="3" gridlines="both" width="100%">
<ItemTemplate>
<table border="0" width="100%">
<tr>
<td>
<%# DataBinder.Eval(Container.DataItem,"category")%>
</td>
</tr>
<tr>
<td>
Active(Yes/No) : <%#IsActive(DataBinder.Eval(Container.DataItem,"active"))%>
<br />
</td>
</tr>
<tr>
<td>
<a href='./more.aspx?id=<%# DataBinder.Eval(Container.DataItem,"id")%>'>more</a>
</td>
</tr>
</table>
</ItemTemplate>
</asp: DataList>
</form>
</body>
</html>
```

في منطقة الأكواد نرى أننا أضفنا دالة أسميناها IsActive من نوع كائن Object نمرر لها بارامبتر يقوم بفحصه فإن كانت قيمته

Trueأظهر الناتج Yes و إلا سيظهر لنا كلمة No

بعد ذلك تأتي أكواد الإتصال في منطقة حدث تحميل الصفحة Page_Load و ربط ال DataList و نلاحظ أننا غيرنا خصائص ال

DataListكالتالي

CODE

```
<asp: DataList id="dl" runat="server" RepeatDirection="Horizontal"
repeaterlayou="table" borderwidth="3" gridlines="both" width="100%">
```

أيضا أعود للدالة و نرى كيف تم إستدعائها مع تمرير بارامينر

CODE

```
<%#IsActive(DataBinder.Eval(Container.DataItem,"active"))%>
```

نرى دائما عند محاولة الكتابة للعناصر التي تخضع لعملية ال Data Binding في منطقة ال HTML نستخدم معها علامة الموجه

Directiveو هي الإشارة # ثم نقوم بكتابة البيانات أو إستدعاء أي دالة

من أهم الخصائص كانت ال RepeatDirection جعلناها Horizontal بدل Vertical التي تكون Default

إستخدام الأنماط Styles مع ال: DataList

BackColor

يتحكم في لون خلفية الكلام الذي سيظهر داخل العنصر.

BorderColor

يظهر لون الحدود الخاصة بالعناصر التي ستظهر.

BorderStyle

نمط أو شكل الحدود التي ستظهر به و هناك عدة قيم بالعادة يتم إستخدامها و هي:

(Dashed-Dotted-Double-Groove-Inset-None-NotSet-OutSet-Ridge and Solid)

BorderWidth

تحدد حجم الحدود التي سنعرض بها العناصر.

CssClass

تحدد نوع الفئة التي سيرتبط بها العنصر من خلال ملف CSS يحوي عدة أنماط على شكل فئات.

Font

تحديد نوع الخط.

ForeColor

لون الخط الذي سنكتب به.

Height

إرتفاع العنصر الذي سيتم عرضه.

HorizontalAlign

تتحكم في الإرتفاع الأفقي للعنصر و القيم المعتاد كتابتها هي:

(Center-Justify-Left-NotSet and Right)

VerticalAlign

تحديد المحاذاة الأفقية للعنصر و القيم المعتاد إستخدامها هي:

(Bottom-Middle-NotSet and Top)

Width

عرض العنصر الذي سيعرض.

كانت هذه معظم الأنماط المستخدمة مع ال.DataList

لنأخذ مثالين صغيرين يقومان بعمل تنسيقات معينة للعناصر:

CODE

```
<asp: DataList
ItemStyle-Font-Italic="True"
ItemStyle-Font-Bold="True"
ItemStyle-Font-Name="Tahoma"
Runat="server"
>

or

<asp: DataList
AlternatingItemStyle-ForeColor="Blue"
AlternatingItemStyle-Font-Name="Arial"
AlternatingItemStyle-Font-Size="14pt"
Runat="server"
>
```

و هكذا رأينا كيف نضيف الأنماط Styles إلى ال DataList كي تظهر لنا البيانات بالشكل الذي نرغب به.

أحداث ال: DataList

1.CancelCommand:

من أجل تفعيل هذا الحدث نقوم بإضافة الخاصية CommandName للزر و نجعل قيمته.cancel

2.DeleteCommand:

نقوم بجعل الخاصية CommandName تساوي.delete

3.EditCommand:

نجعل قيمة ال CommandName هي.edit

4.UpdateCommand:

نجعل قيمة ال CommandName هي.update

5.ItemCommand:

يتم تفعيل هذه الخاصية عند وجود حدث غير ما ذكرته بالأعلى من حذف و تعديل و إلغاء و إدخال.

و لكن ما هي ال CommandName أو بشكل آخر أين تكتب؟؟

الجواب سهل جدا و هي خاصية تضاف لأحد الأدوات التالية:

Button

LinkButton

ImageButton

نضيف لهم الخاصية CommandName بحيث عند الضغط على أحدهم يتم تنفيذ الحدث المعين مع العلم أنه يجب إضافة الحدث لل DataListو هذا سيتضح لنا من خلال الأمثلة القادمة.

لنأخذ مثالا نستخدم فيه أغلب الأحداث بشكل بسيط بحيث نحتاج إلى DataList و Label تقوم بكتابة جملة تدل على الحدث الذي إخترناه

CODE

```
<%@ Page Language="C#" Debug="True" %>
<%@ import Namespace="System.Data.OleDb" %>
<script runat="server">

void dataBind(){
string prov="Provider=Microsoft.Jet.OleDb.4.0;Data Source=" +
Server.MapPath("db.mdb");
string SQL="SELECT * FROM Categories";
OleDbConnection cn = new OleDbConnection(prov);
OleDbCommand cmd = new OleDbCommand(SQL,cn);

cn.Open();
OleDbDataReader reader = cmd.ExecuteReader();
dl.DataSource= reader;
dl.DataBind();
```

```
cn.Close();
}
void Page_Load()
{
if(!Page.IsPostBack) {
dataBind();
}
}

void dl_ItemCommand(Object source, DataListCommandEventArgs e) {
msg.Text="<li>Item Command!";
}

void dl_EditCommand(Object source, DataListCommandEventArgs e) {
msg.Text="<li>Editing Item!";
}

void dl_DeleteCommand(Object source, DataListCommandEventArgs e) {
msg.Text="<li>Deleting Item!";
}

void dl_UpdateCommand(Object source, DataListCommandEventArgs e) {
msg.Text="<li>Updating Item!";
}

</script>
<html>
<head>
</head>
<body>
<form runat="server">
<p>
<asp : DataList id="dl" runat="server" OnUpdateCommand="dl_UpdateCommand"
OnDeleteCommand="dl_DeleteCommand" OnEditCommand="dl_EditCommand"
OnItemCommand="dl_ItemCommand" repeatColumns="1" repeaterlayout="table"
ItemStyle-Font-Name="tahoma" borderwidth="1" gridlines="both" width="33%">
<ItemStyle font-names="tahoma"></ItemStyle>
<ItemTemplate>
<table border="0" width="100%">
<tr>
<td>
<%# DataBinder.Eval(Container.DataItem,"category")%>
<br />
<asp:LinkButton runat="server" Text="Edit!" CommandName="edit" />
<asp:LinkButton Text="Delete!" CommandName="delete" runat="server" />
<asp:LinkButton Text="Update!" CommandName="Update" runat="server" />
</td>
</tr>
</table>
</ItemTemplate>
</asp : DataList>
</p>
<asp:Label id="msg" runat="server" text=""></asp:Label>
</form>
</body>
```

```
</html>
```

نبدأ في تحليل المكتوب في المثال..

في الأعلى نرى أكواد الإتصال بقاعدة البيانات و هذه مألوفة لنا و بعدها هناك بعد الأحداث نعود لها بعد قليل.

نذهب لجزء ال HTML حيث الأدوات و قلنا أننا نحتاج ل Label نسميه msg و DataList نسميها dl كالتالي:

CODE

```
<asp : DataList id="dl" runat="server" OnUpdateCommand="dl_UpdateCommand"
OnDeleteCommand="dl_DeleteCommand" OnEditCommand="dl_EditCommand"
OnItemCommand="dl_ItemCommand" repeatColumns="1" repeaterlayout="table"
ItemStyle-Font-Name="tahoma" borderwidth="1" gridlines="both" width="33%">
```

نرى الخصائص التي أضفناها و الأحداث التي عرفناها لتكون متصلة بال DataList مثلا لإضافة الحدث الخاص بالحدث نضيف هذه

الخاصية:

CODE

```
OnDeleteCommand="dl_DeleteCommand"
```

قمنا بعد ذلك بتحديد ال ItemStyle من أجل تنسيق النمط الخاص بالمخرجات و أكتفيت فقط بجعل الخط Tahoma و من ثم أضفنا

الوسم ItemTemplate كما هو موضح هنا:

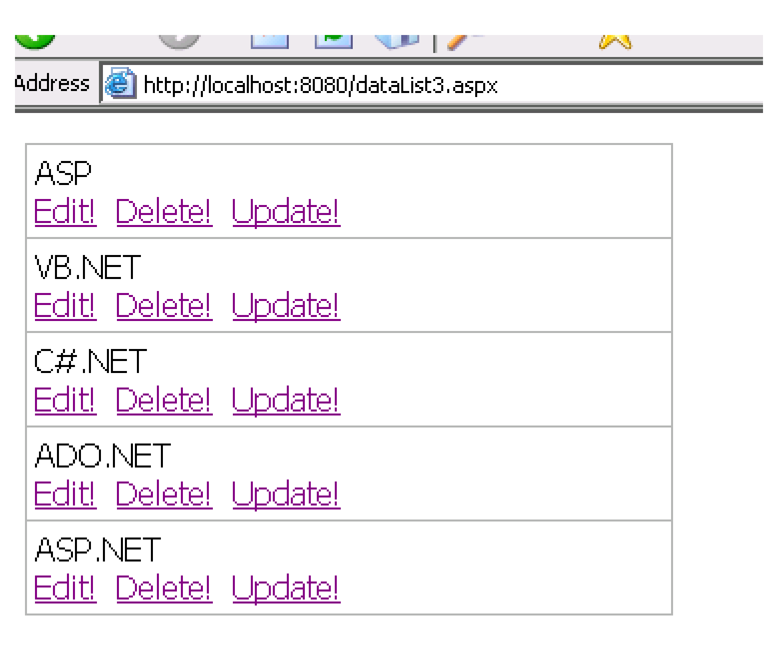
CODE

```
<ItemTemplate>
<table border="0" width="100%">
<tr>
<td>
<%# DataBinder.Eval(Container.DataItem,"category")%>
<br />
<asp:LinkButton runat="server" Text="Edit!" CommandName="edit" />
<asp:LinkButton Text="Delete!" CommandName="delete" runat="server" />
<asp:LinkButton Text="Update!" CommandName="Update" runat="server" />
</td>
</tr>
</table>
</ItemTemplate>
```

طبعا أضفنا جدولا من أجل تنسيق المخرجات في كل صف سيظهر لنا إسم التصنيف Category و تحته ٣ أزرار من نوع

LinkButtonو كل واحد منها أضفنا له الخاصية CommandName و أسندنا لها القمية المطلوبة سواء حذف أو تعديل أو

إدخال كما هو موضح بهذا الشكل:



طبعا ستظهر عدة بيانات أو أقسام على حسب السجلات الموجودة في جدول البيانات و مع كل سجل ستظهر ٣ أزرار كل منها عبارة عن رز يقوم بإستدعاء حدث معين و لنفرض أننا ضغطنا الزر الذي يظهر لنا الكلمة !Delete فماذا سيحدث؟

سيتم إستدعاء الحدث dl_DeleteCommand حسب ما أضفنا لخصائص ال DataList و دعونا نعود لمنطقة الأكواد و لنرى هذا الحدث ماذا يحتوي:

CODE

```
void dl_DeleteCommand(Object source, DataListCommandEventArgs e) {
msg.Text="<li>Deleting Item!";
}
```

فقط أضفنا جملة تقوم بجعل ال Label يقوم بعرض جملة مكتوب بها !Deleting Item كي نرى ما هو الزر الذي ضغطنا عليه و لكن هي كانت فكرة كي نرى كيف يتم إستدعاء الأحداث و لعل الأمثلة القادمة تزيد الإثارة و التعرف أكثر على الأحداث.

الباب الثانى

كيف تصنع ويب سيرفز من خلال برنامج Web Matrix

خدمة الإنترنت ASP.NET Web Service:

هي عبارةٌ عن برنامجٌ يعملُ على خوادم الإنترنت Internet Servers لاستقبال الطلبات، تمامًا مثل تطبيق الإنترنت، ولكن دون أن يرد بصفحةٍ من صفحات الإنترنت، بل يرسل ناتج العملية الحسابية، أو المعلومات المطلوبة من قاعـدة بياناتٍ ما، وعادةً ما تُطلب هذه الخدمات من قبل خادم إنترنت آخر، حيث سيتولّى هو التعامل مع البيانات.

ولتوضيح الفارق نضربُ هذا المثال:

إن تطبيق الإنترنت الذي يتلقى منك استعلامًا عن كل كتب (البرمجة) سيعرض لك صفحةً عليها أسماء هذه الكتـب..
أما خدمة الإنترنت التي تتلقى منك نفس هذا الاستعلام، فسترسل إليك ملف XML يحتـوي علـى أسـماء الكتـب المطلوبة، حيثُ يمكن لتطبيقك في هذه الحالةِ أن يتعامل مع البيانات ويعرضها في صفحةٍ جديدةٍ بالشكل الذي تريـده، ويرسلها للمستخدم الذي تريده، أو ربما تعرضها على نموذج ويندوز عادي.

الان اتبع الخطوات التالية لنتعلم كيف نصنع ويب سيرفز:

١- افتح برنامج **Microsoft ASP.NET Web Matrix ستري مربع حوار يسأل على اسم الملف الجديد و اللغة المستخدمة**
٢- اختر **(General) من تبويب Templates**
٣- اختر **XML Web Service و اكتب مسار الملف الذى سنقوم بعملة فى خانة Location**
٤- G:\webmatrix28-6\lessons
٥- **اكتب** *myFirstWebService.asmx فى خانة* **Filename**
٦- **اختر اللغة Visual C# من قائمة** the **Language** drop-down box
٧- اكتب *Sample فى خانة* the **Class** box
٨- اكتب *MatrixDemo فى خانة* **Namespace** box
٩- انقر بعد ذلك على ok

شاهد الشكل التالى



فى مثالنا التالى سوف نقوم بعمل WebMethod جديدة عبارة عن ناتج طرح رقمين وواضح ان الناتج سيكون Integer

١٠- افتح myFirstWebService.asmx فى نافذة الكود Code view

١١- سوف نقوم بتسمية الwebmethod باسم Add و نقوم باضافة الكود اليها حيث المدخلات عبارة عن متغيرين رقمين

و المخرجات هى ناتج الجمع متغير رقمى ايضا أى ان كل المتغيرات من النوع type Integer

١١- أكتب الكود التالى

```
<% @WebService language="C#" class="Sample "%>

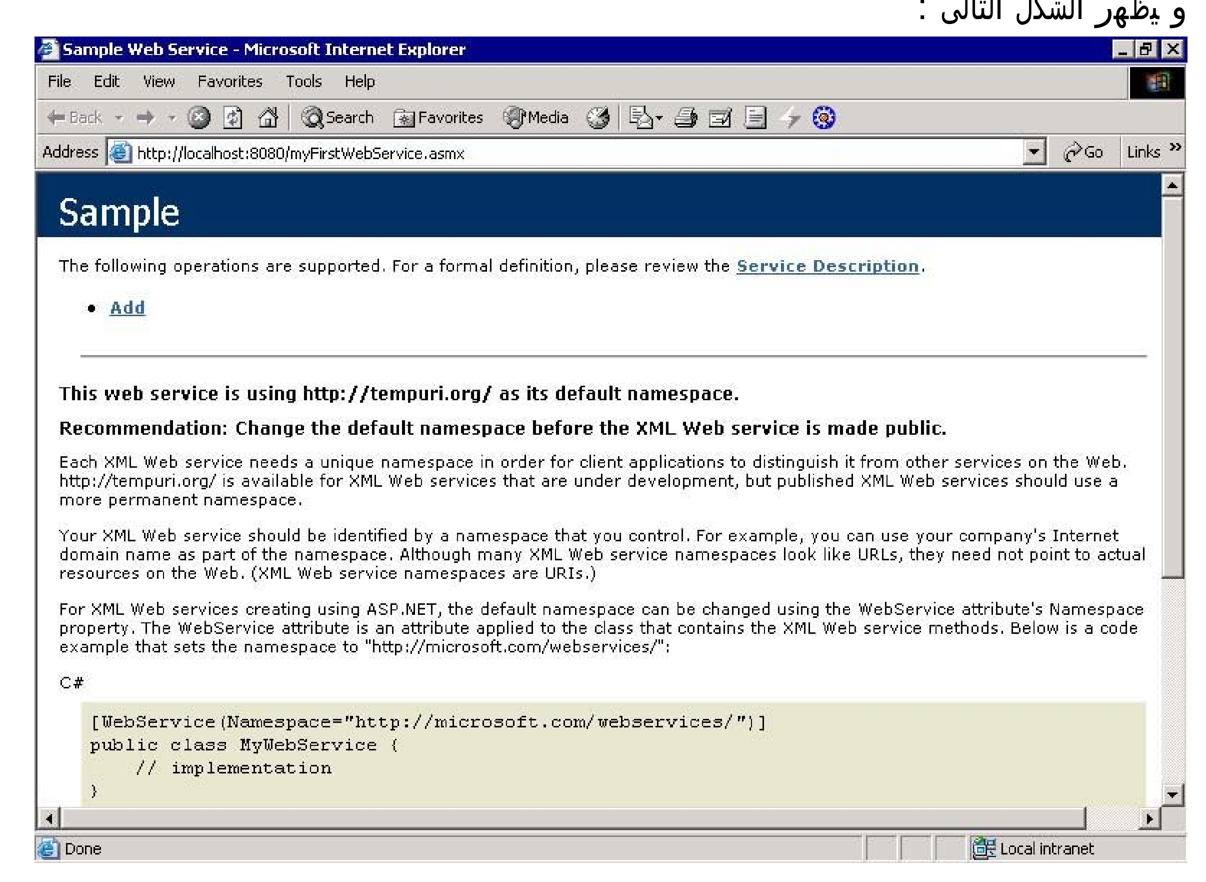
using System;
using System.Web.Services;
using System.Xml.Serialization;
```

```
public class Sample
{

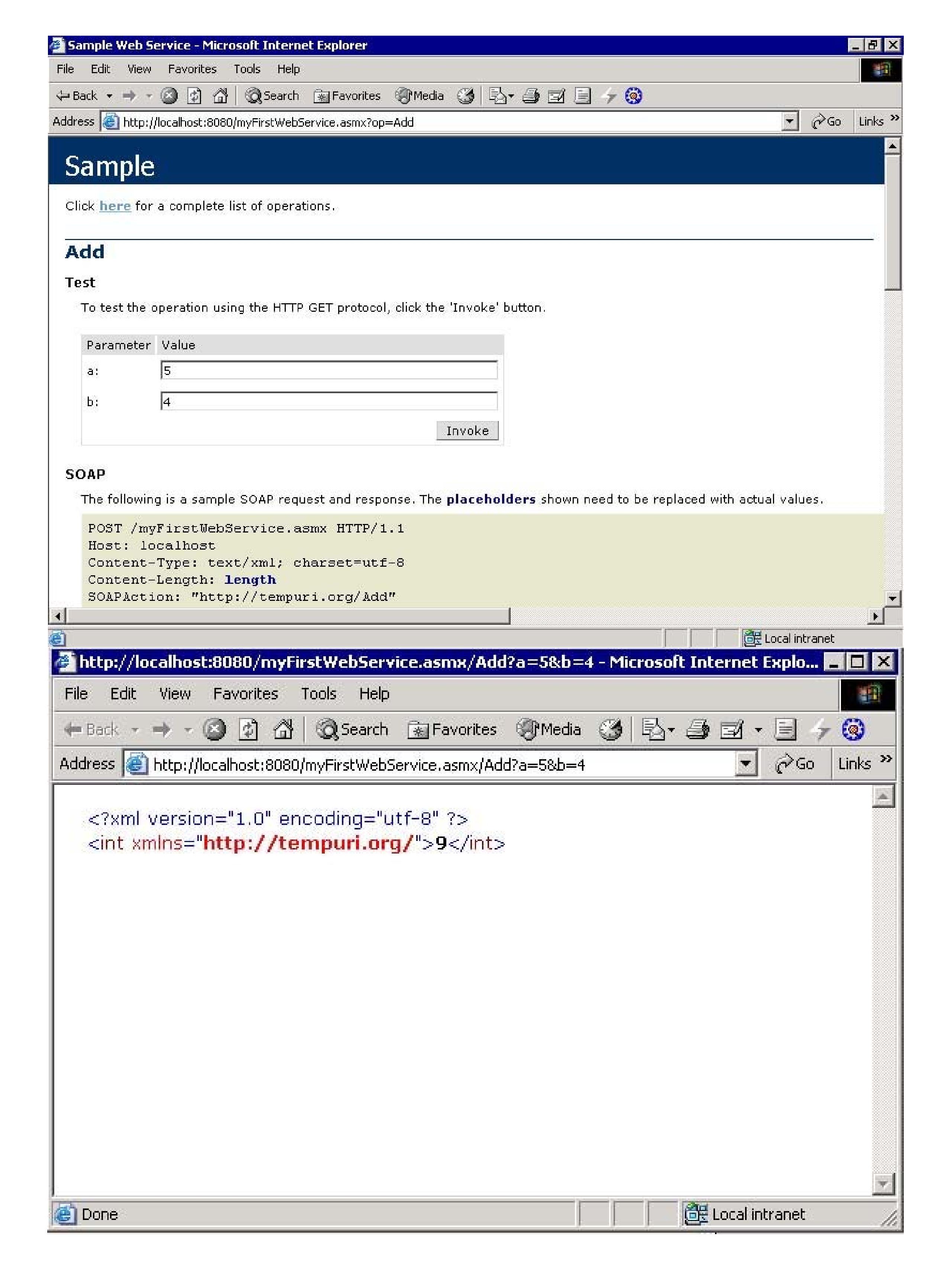
[WebMethod]
{
public int Add(int a, int b  )
return a + b;
}
}
```

١٢- احفظ الملف ثم قم بالضغط على مفتاح F5 ، يقوم البرنامج بفتح نافذة جديدة من برنامج انترنت اكسبلور

و يظهر الشكل التالى :



١٣- يظهر فى الصورة الويب سيرفز التى قمت بعملها و اسمها Add قم بالنقر عليها لكى ترى اختبار لما فعلته schema details ثم قم بكتابة رقم ٥ فى الخانة a ثم رقم ٤ فى الخانة b ثم انقر على Invoke فيظهر حاصل الجمع ٩ كما فى الشكل التالى



١٤- الان قد صنعت اول ويب سيرفز خاصة بك تجمع رقمين.

١٥- و لكن نريد ان نقوم بعمل ويب سيرفز نستخدمها مع قاعدة بيانات الاكسيس لكى نتصل بها و نعرض البيانات من جدول ما فى قاعدة البيانات من خلال DataSet

عمل ويب سيرفز نستخدمها مع قاعدة بيانات الاكسيس

لذلك فى البداية سوف نقوم بعمل قاعدة بيانات ببرنامج الاكسيس نذهب للقرص الصاب فى المجلد G:\webmatrix28-6\lessons

نضع به قاعدة بيانات باسم db من نوع أكسيس بها جدول واحد إسمه Categories و به ٣ حقول كالتالي:

1 : id هو المفتاح الوحيد PrimaryKey من نوعAutoNumber

2 : Category من نوع Text

3 : Active من نوع Boolean و هي بالأكسيسYes/No

نقوم بكتابة عدد من الأقسام Categories في الجدول ثم نحفظ البيانات

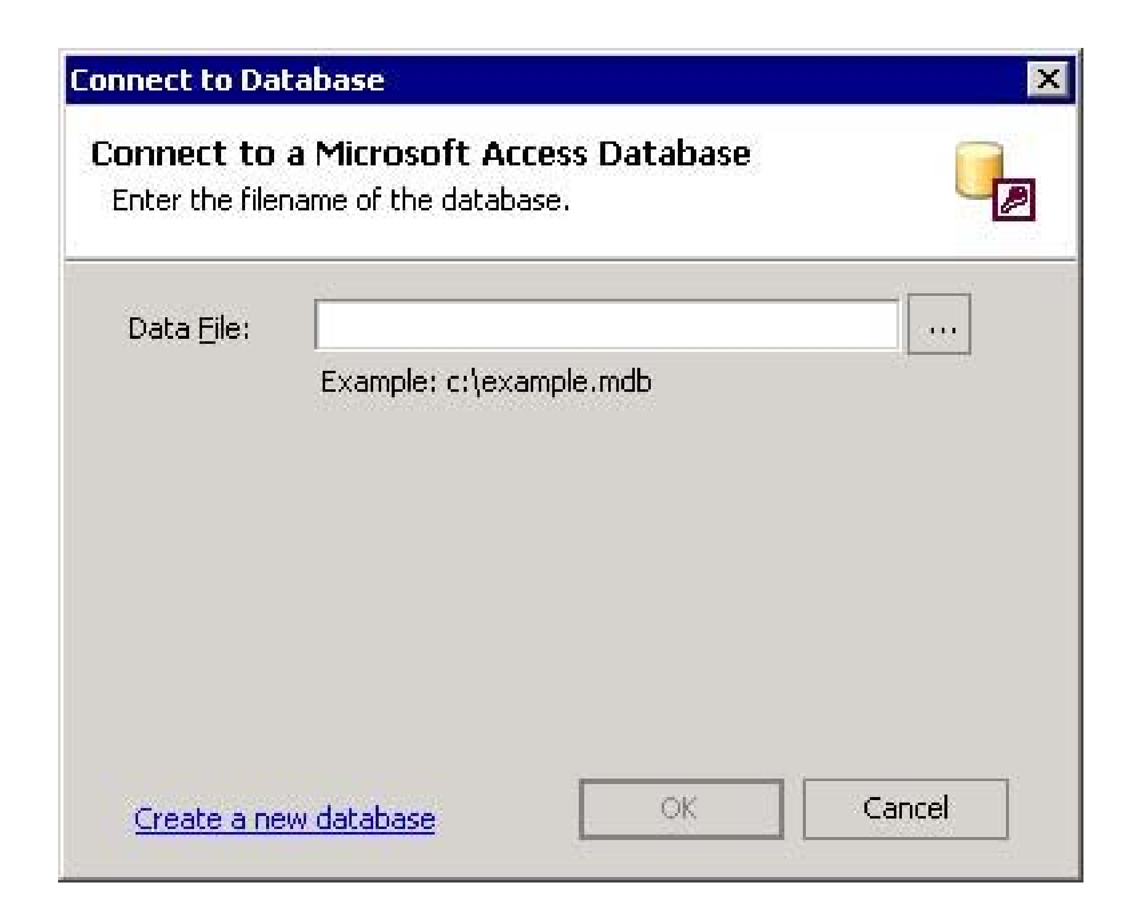
و نغلق ال MS Access و نتوجه للعمل برمجيا مع ال.MS WebMatrix

١٦- من التبويب the Code Builders من قائمة Toolbox اسحب الاداة SELECT Data Method Code Builder و قم بأدراجها

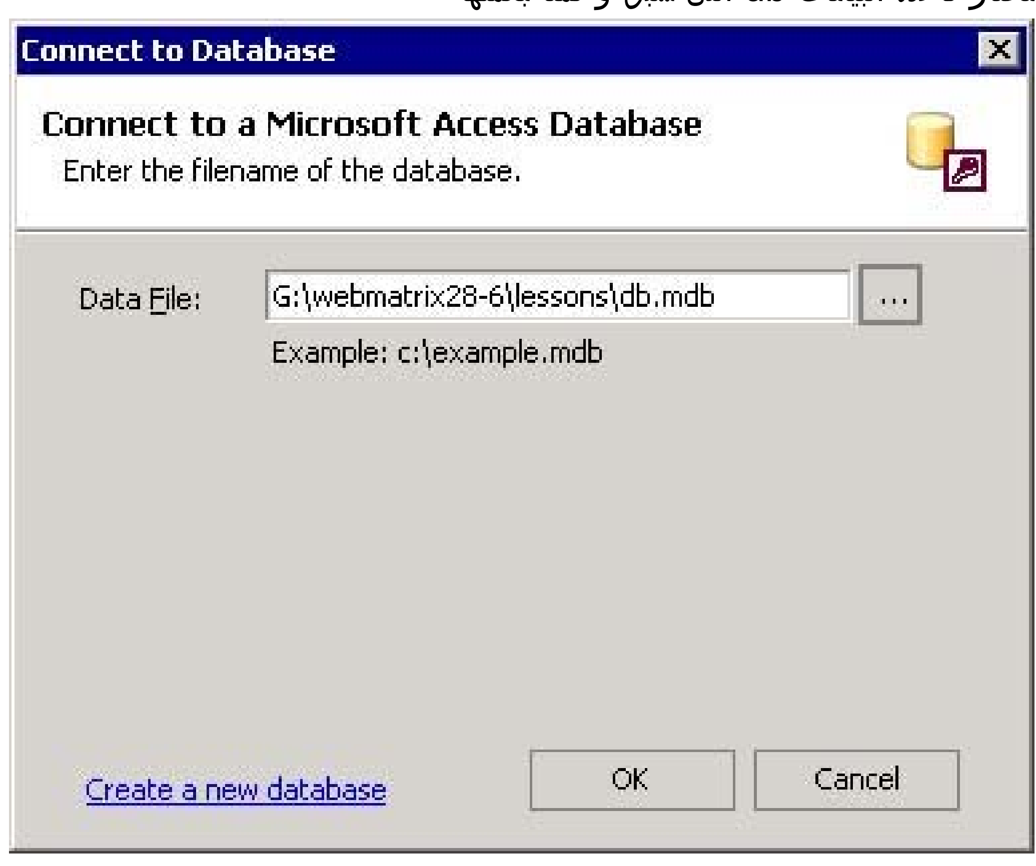
داخل ملف العملmyFirstWebService.asmx بعد القوس قبل الاخير للويب سيرفز السابقة Add فيظهر لك الشكل التالى



١٧- اضغط على على Create فيظهر لك مربع حوار لتختار منة اسم قاعدة البيانات و مكانها على القرص الصاب



تختار قاعدة البيانات db التى سبق و قمنا بعملها



١٨- انقر على ok فيظهر لك معالج الاستعلامات

SELECT Data Code Wizard

Construct a SELECT query

Check the columns you want returned and build the WHERE clause.

Tables:

Categories

Columns:

*

id

Category

Active

Select All

Select None

WHERE clause:

WHERE

OR Clause

Delete

Preview:

SELECT FROM

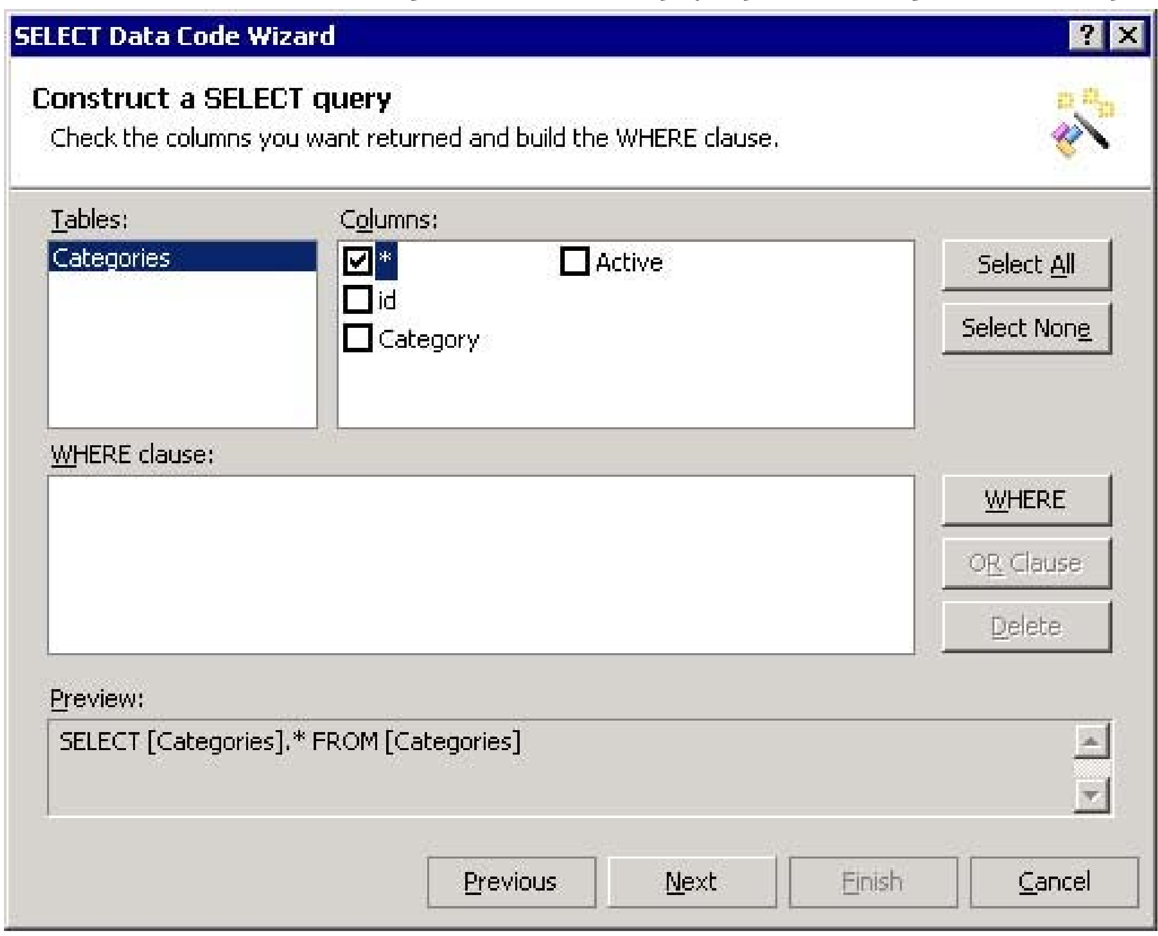
Previous

Next

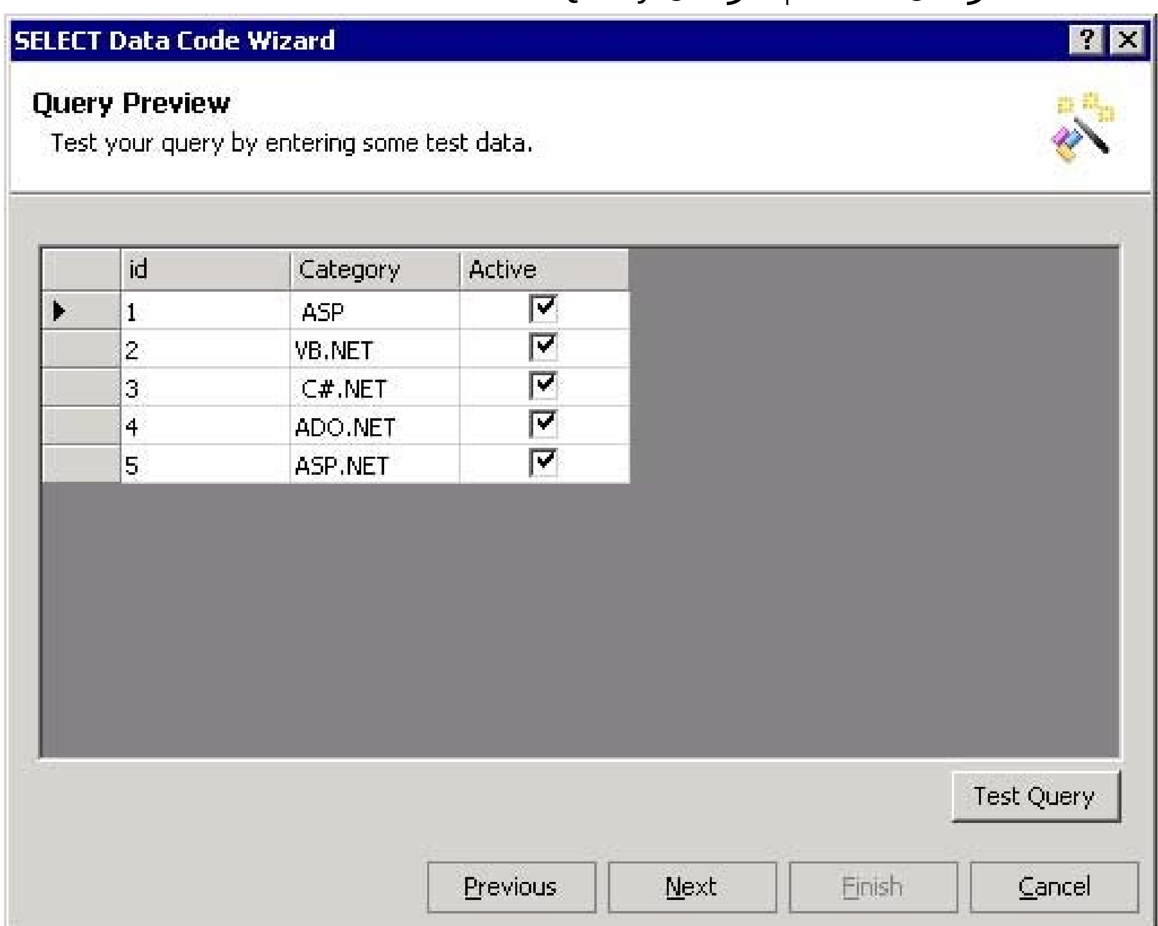
Finish

Cancel

انقر على علامة * و معناها اختيار اظهار كل الاعمدة من جدول Categories



١٩- انقر على Next ثم انقر على Test Query



٢٠- كما فى الشكل يظهر كل محتويات الجدول انقر بعد ذلك على Next

SELECT Data Code Wizard

Name Method

Enter a name for the method to be generated.

This CodeBuilder will generate a public method with strongly-typed parameters. What should the method be called?

Getorders

The method can return either an System.Data.SqlClient.SqlDataReader or a System.Data.DataSet. Which do you want?

- (•) DataSet
- ( ) DataReader

Previous | Next | Finish | Cancel

٢١- اترك الاختيارات كما هى لكن قم بتسمية الMethod اسم مناسب و ليكن Getorders و لاحظ انك اخترت ان تكون النتيجة ترجع فى Dataset ثم فى النهاية انقر على Finish

٢٢- ينتهى المعالج الالى المدمج فى البرنامج من اعداد ال Method و يقوم بادراج الكود التالى فى الملف

```
<% @WebService language="C#" class="Sample "%>

using System;
using System.Web.Services;
using System.Xml.Serialization;

public class Sample
{

[WebMethod]
{
public
int Add(int a, int )   ;
 {
return a + b   }

[WebMethod]
public
{  System.Data.DataSet GetOrders()
string connectionString = "Provider=Microsoft.Jet.OLEDB.4.0; Ole DB Services=-4; Data
Source=G:\\ webmatrix28\\ lessons\\db.mdb ";
System.Data.IdbConnection dbConnection = new
System.Data.OleDb.OleDbConnection(connectionString);
```

```
string queryString = "SELECT [Categories]. * FROM [Categories]";
System.Data.IdbCommand dbCommand = new
System.Data.OleDb.OleDbCommand();
dbCommand.CommandText = queryString    ;
dbCommand.Connection = dbConnection  ;
System.Data.IdbDataAdapter dataAdapter = new
System.Data.OleDb.OleDbDataAdapter ();
dataAdapter.SelectCommand = dbCommand   ();
System.Data.DataSet dataSet = new System.Data.DataSet   ();
(dataAdapter.Fill(dataSet    );

return dataSet   ;
}
}
```

٢٣- قم بحفظ الملف ثم اضغط F5 لتجربة العمل ستظهر لك الشاشة التالية

Start Web Application

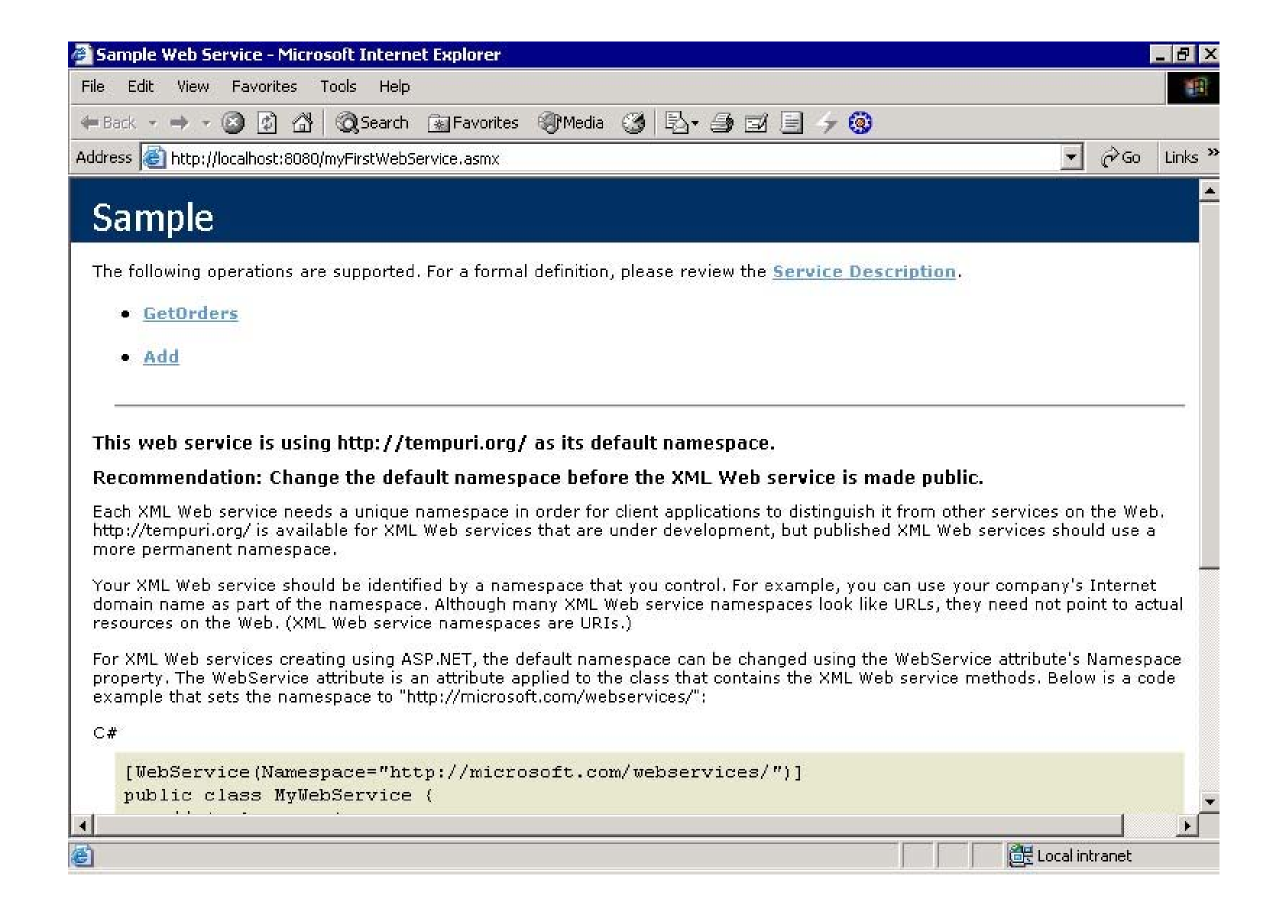
Start Web Application
Start a Web application at the selected application directory.

Application Directory: G:\webmatrix28-6\lessons

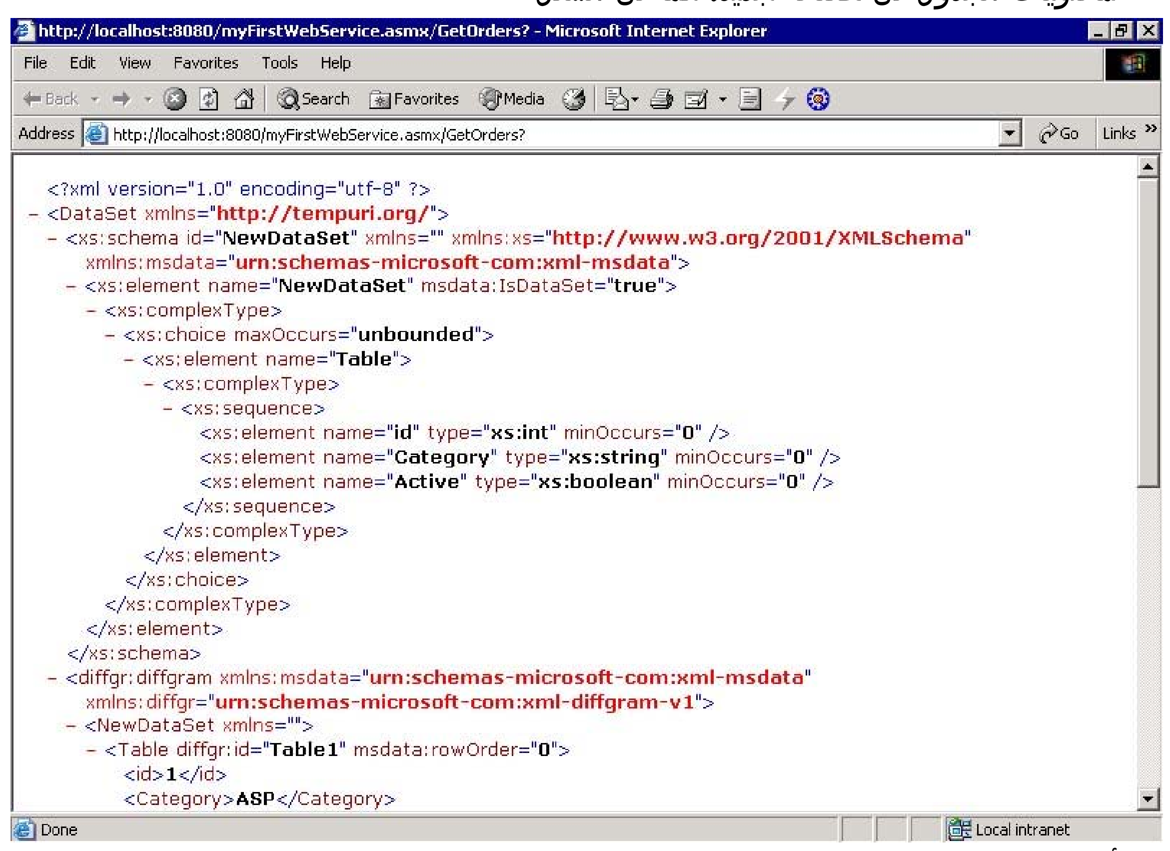
Use ASP.NET Web Matrix Server
Application Port: 8080
Use or create an IIS Virtual Root
Application Name:
Enable Directory Browsing

Start | Cancel

انقر على Start فتظهر لك نافذة جديدة



٢٤- انقر على Getorders تظهر صفحة جديدة انقر على Invoke ستظهر لك نتيجة العمل و هى محتويات الجدول فى صفحة جديدة كما فى الشكل



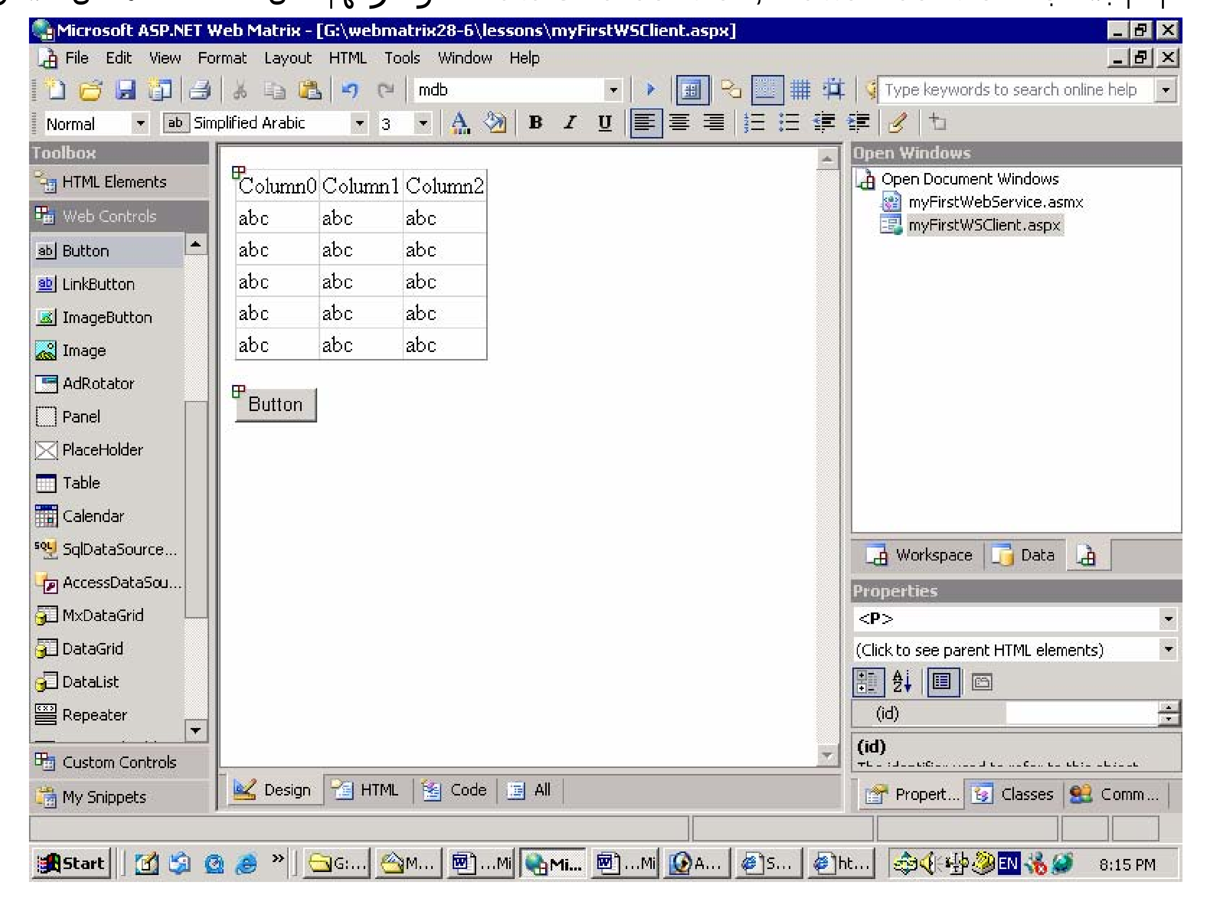
قم بأغلاق النافذة و لاحظ دائما انة يظهر فى اسفل الشاشة الشكل التالى عند الضغط على F5 ثم start



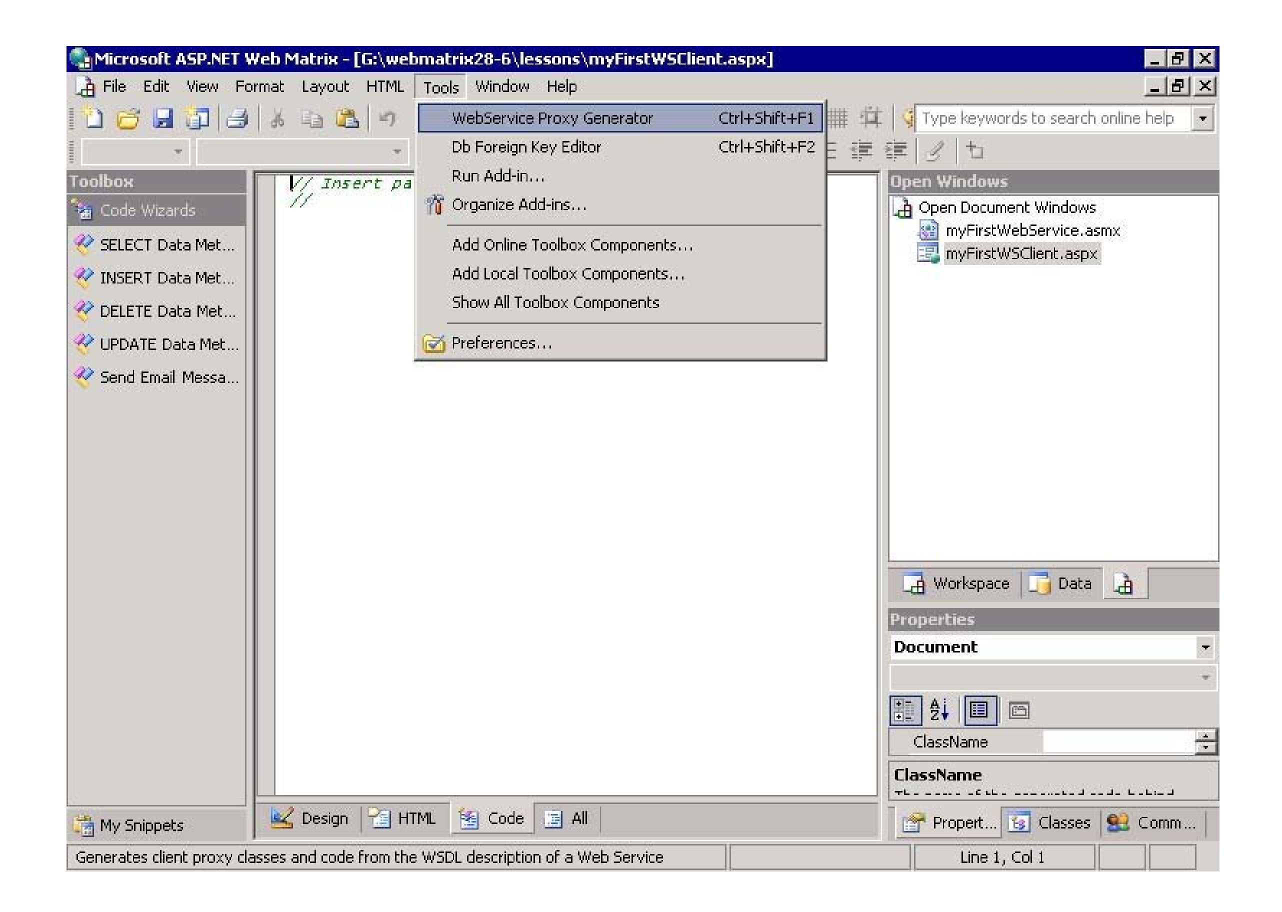
و هذا يعنى ان السيرفر المدمج مع البرنامج Web Matrix يعمل بشكل جيد .

٢٣- الان تسأل ماذا سأفعل بهذة الويب سيرفز او كيف يمكننى استخدامها من خلال ملف aspx عادى ان الاجابة على هذا السؤال هى الهدف من كل الدرس لذلك سوف نقوم بعمل ملف جديد اسمة myFirstWSClient.aspx و اللغة المستخدمة لكتابة الكود هى C# و احفظة مع الملف السابق فى نفس المجلد على القرص الصلب

ثم قم بسحب DataGrid control , Button control و اتركهم على الصفحة كما فى الشكل التالى :



٢٥- اضغط على التبويب Code ثم انقر على Tools من شريط القوائم بأعلى البرنامج و اختر من القائمة المنسدلة WebService Proxy Generator كما فى الشكل التالى :



يظهر مربع حوار ادخل فية البيانات التالية MatrixDemo فى مربع Namespace box و http://localhost:8080/myFirstWebService.asmx فى مربع WSDL URL box فيصبح مربع الحوار كما فى الشكل التالى

انقر على Generate فيظهر لك الرسالة التالية عندما ينجح فى عمل الاتصال مع الويب سيرفز



انقر على ok ثم افتح ملف myFirstWSClient.aspx فى وضع التصميم Design و انقر مرتين على الزر الذى سبق انشأتة Button1

فتظهر نافذة كتابة الكود فقم بعمل كائن من الويب سيرفز التى سبق ان صنعتها من خلال الكود التالى

```
MatrixDemo.Sample  wsProxy = new MatrixDemo.Sample();
```

ثم قم بتعبئة الDataGrid1 بالDataSource و هنا فى مثالنا هو الGetOrders()

فنكتب الكود كالتالى

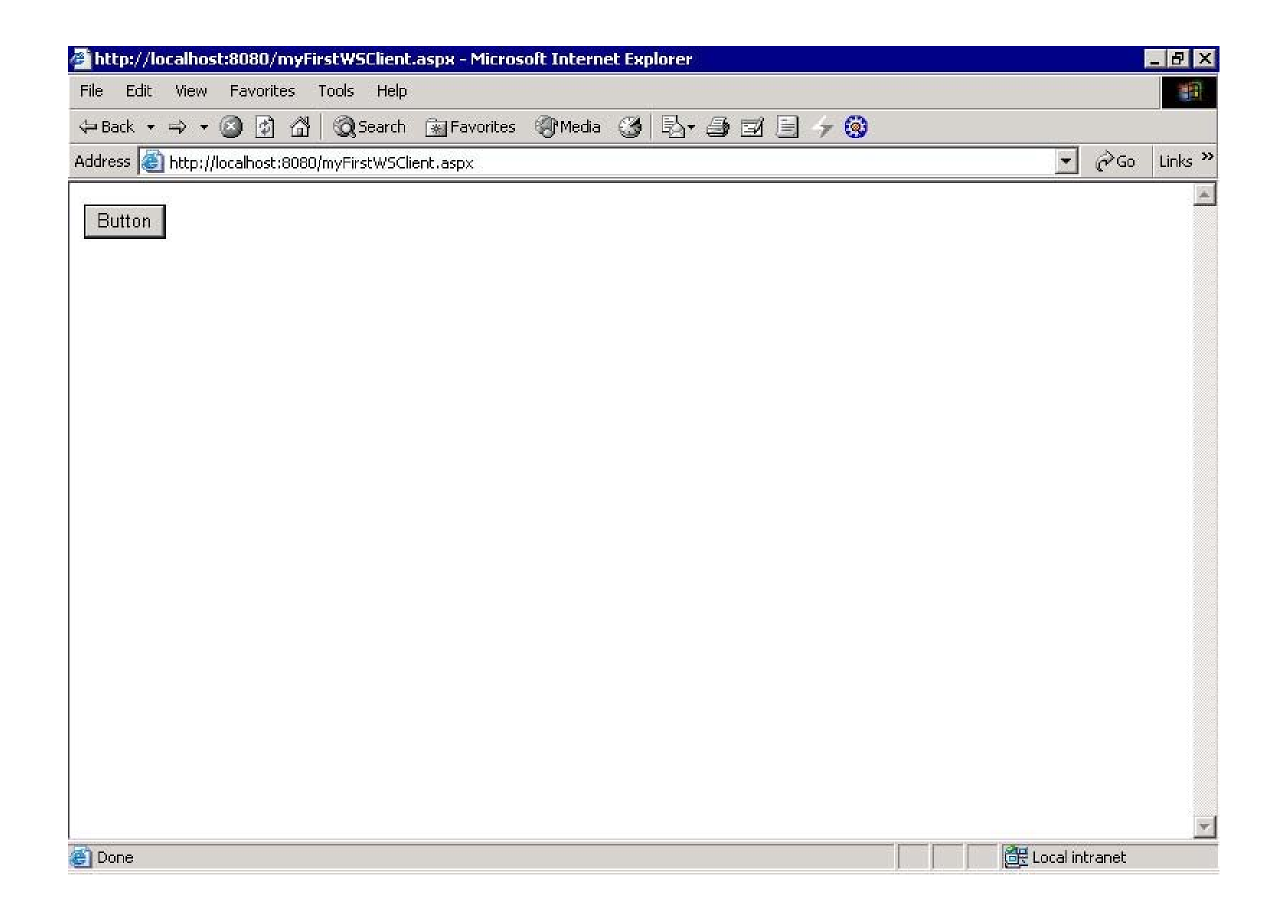
```
DataGrid1.DataSource = wsProxy.GetOrders();
DataGrid1.DataBind();
```

الكود كلة داخل حدث النقر على الزر هكذا

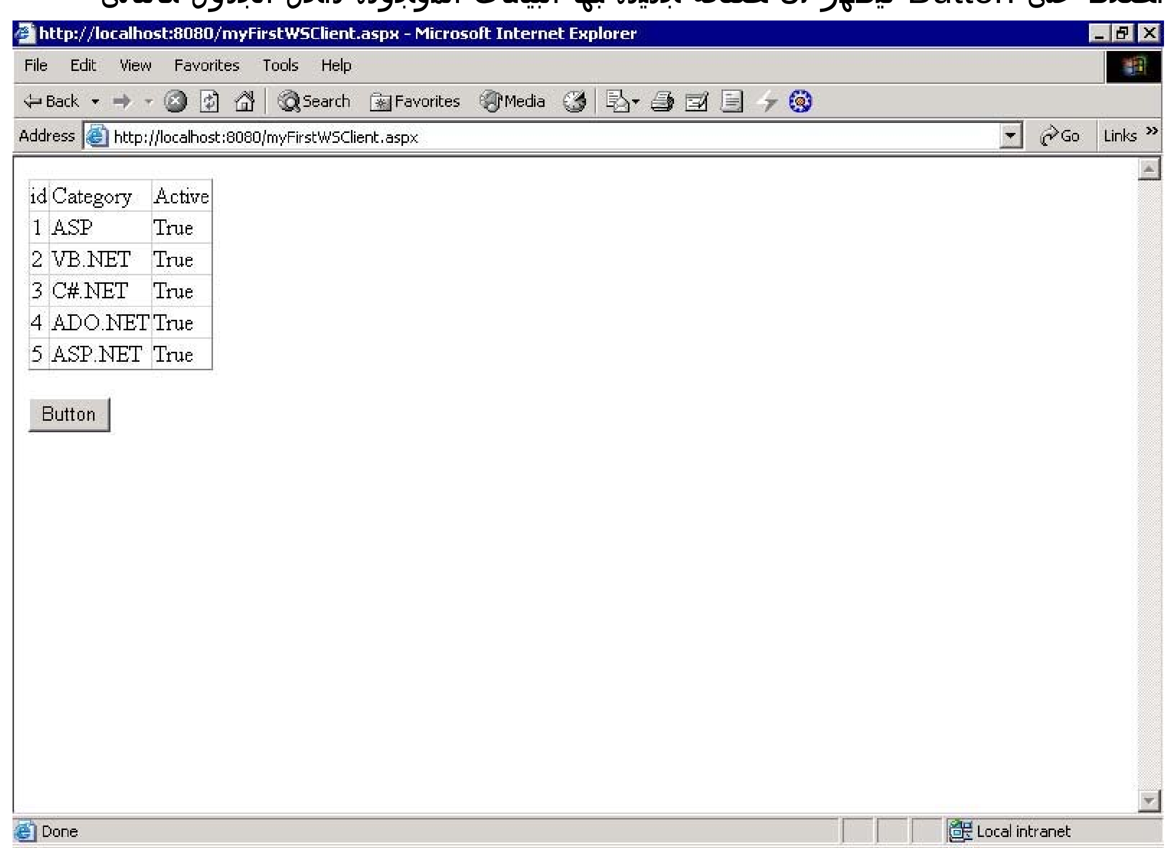
```
void Button1_Click(object sender, EventArgs e)
{

MatrixDemo.Sample  wsProxy = new MatrixDemo.Sample();
DataGrid1.DataSource = wsProxy.GetOrders();
DataGrid1.DataBind();
}
```

٢٦- اخيرا قم بتجربة العمل من خلال F5 ثم Start فتظهر لك الصفحة التالية



اضغط على Button فيظهر لك صفحة جديدة بها البيانات الموجودة داخل الجدول كالتالى



id	Category	Active
1	ASP	True
2	VB.NET	True
3	C#.NET	True
4	ADO.NET	True
5	ASP.NET	True

اعتقد اننا الان نعرف كيف نتعامل مع ويب سيرفز

انتهت الفصول الاولى من الكتاب و أرجو ان يعجبكم و اعدكم بأستكمال الكتاب بموضوعات متقدمة و شرح افضل للأمكانيات لغة ال c# و ارجو المعذرة لتقصيرى و اهتمامى بشرح امكانيات برنامج Webmatrix أكثر من توضيح الكود و لكن يمكنكم الرجوع لقائمة المراجع فستجدون المواقع التى اقتبست منها بعض الاكواد و الشرح ربما تفيدكم كما استفدت منها كثيرا و رجاء فى حالة وجود خطأ او استفسار مراسلتى على بريدى الالكترونى

و شكرا .

الكاتب : ما يكل نبيل أخنوخ

بريد الكترونى ramegypt@yahoo.com

www.ramegypt.cjb.net

المراجع :

منتديات الفريق العربي للبرمجة

مقالات متعددة للعضو المتميز محمد خليل أبو صلاح

موقع WWW.ASP.NET

http://www.microsoft.com/

ملحق عن برنامج Access 2000 و قواعد البيانات

ما هي قاعدة البيانات؟:

قاعدة البيانات Database هي تركيبٌ تستطيع من خلاله تخزين كميةً ضخمةً من البياناتِ التي تربطهـا علاقـات معينة، مثل الجداول.. وحقيقةُ الأمر، فإن قاعدة البيانات ما هي إلا ملفّ File أو مجموعة ملفّات تُخزن على القرص الصلب Hard Disk ككلّ الملفّات العادية، ولكن ما يميزها عن باقي الملفّات، هو قدرتك على كتابة البيانـات فيهـا وقراءتها منها، باستخدام الوسائل التي تمنحها لك برامج قواعد البيانات، دون الحاجة إلى التعامل مع الملفّ مباشـرةً، وهو ما يجعل الأمر أكثر سهولةً وتنظيمًا بالنسبة لك.

ولا مانع أن تخزن بياناتك في ملفّ خاص بك بالطريقة التي تريحك، ولكن استخدام قواعد البيانات يوفّر لك الكثير من الوقتِ والجهد.

وهناك أنواع كثيرة من قواعد البيانات، تبعا للشركة التي تنتجها والبرامج التي تنشئها، فهناك قواعد بيانـات برنـامج Access وبرنامج SQL Server وهما من إنتاج ميكروسوفت، وهناك عشرات البرامج غيرهما من إنتاج شركات أخرى.

وتسمى هذه البرامج "أنظمة إدارة قواعد البيانات" Database Management Systems (DBMS)، ومهمتها الأساسية أن تمنحك الوسائل والأدواتِ اللازمة لإنشاءِ قواعِد البياناتِ والتعامل معها، بأسهل طريقةٍ وأفضل إمكانيات، بحيثُ تقوم بدور الوسيط بينك وبين البيانات المخزنة في ملفّ قاعدة البيانات.

ومهما كان نوع قاعدة البيانات التي تتعامل معها، ومهما كانت طريقة تخزينها في الملفّ، فإن كلّ قواعد البيانات تتبع قواعد أساسية وتحقّق شروطا معينة متعارفا عليها دوليا، كما أنّها كلّها بلا استثناء تستخدم "لغة الاستعلام المركّبة" Structured Query Language (SQL)، وهي لغة خاصة لحفظ واسترجاع وتحديث البيانات في قواعد البيانات.

تركيب قواعد البيانات:

تتركّب قاعدة البيانات من مجموعةٍ من الجداول، هي التي يتم تخزين البيانات بها.. وطبعا أنت تعرف أن الجدول يتكون من أعمدة Columns (وأحيانا تسمى حقول Fields) وصفوف Rows (وأحيانا تسمى سجلات Records).. فمثلا، لو لديك قاعدة بيانات لمكتبة، فلا بد أنّها ستحتوي على جدولٍ يحتوى على عناوين الكتب الموجودة في المكتبة (وليكن اسمه Titles)، وجدولٍ آخر يحتوى على أسماءِ مؤلّفي هذه الكتب Authors، وجدولا ثالثا يحتوى على أسماء الشركات التي نشرت هذه الكتب Publishers.

وفي جدول عناوين الكتب، يمكن إنشاء عدة أعمدة لتخزين بعض المعلومات عن كلّ كتاب، مثل اسمه، وعدد صفحاته، وموضوعه... إلخ.

ولكّنا لن نضع اسم مؤلّف الكتاب في جدول عناوين الكتب، وذلك لسبب جوهري: ذلك أنّه من الممكن أن يحتوى الجدول على أكثر من كتابٍ لنفس المؤلّف.. في هذه الحالة سيتم تكرار اسم المؤلّف أكثر من مرة، مما سيعمل على زيادة حجم قاعدة البيانات، وإبطاء عمليات البحث.. إذن ما الحلّ؟

الحلّ هو تخصيص جدولٍ للمؤلّفين، هذا الجدول يحتوي على رقم مسلسل ID لكلّ مؤلّف.. وفي جدول الكتب ننشئ عمودا نضع فيه رقم مؤلّف الكتاب.. فمثلا لو كان رقم (توفيق الحكيم) في جدول المؤلّفين هو ١٠٠، فإن خانة رقم المؤلّف في جدول العناوين لا بد أن تكون ١٠٠ لكلّ من الكتب التالية "شهرزاد" و"الأيدي الناعمة" و"أرني ا " و"شجرة الحكم".

ولكي تستوعب مقدار التوفير في الحجم، يجب أن تعرف أن كلّ حرف في النص يخزن في وحدة واحدة Byte، مما يعني أن النص "توفيق الحكيم" يحتاجُ إلى ١٢ وحدة تخزين، وتكراره أربع مرات يعني أنّه يحتاج إلى ٤٨ وحدة!.. بينما تخزين الأرقام رقم يصل إلى ٣٢٠٠٠ لن يحتاج إلا إلى وحدتين فقط، مما يعني أن تخزين رقم ١٠٠ أربع مرات يحتاج إلى ٨ وحدات فقط.. طبعا الفرق واضح.

وليتَ الأمر يقتصر على هذا.. إن إنشاء عمود يحتوي على أسماء المؤلّفين يقتضي تصميمه ليستوعب أطول اسم من هذه الأسماء.. افترض أن أطول اسم لمؤلّف هو ٢٠ حرفا، في هذه الحالة لو كان جدول الكتب يحتوي على ٣٠ ألف كتاب، فإن هذا معناه أن عمود أسماء المؤّلفين يحتاج إلى ٣٠٠٠٠ × ٢٠ = ٦٠٠ ألف وحدة، أي ٦٠٠ كيلو بايت!

بينما لو استعضنا عن ذلك بعمود أرقام المؤلّفين، فإنّه سيحتاج إلى ٦٠ ألف وحدة، أي ٦٠ كيلو بايت فقط.

طبعا ستتساءل: ولكنّنا في كلّ الأحوال سننشئ عمودا لأسماء المؤلّفين بالإضافة لعمود رقم المؤلّف في جدول المؤلّفين.. نعم، ولكن عدد المؤّلفين قد لا يزيد عن ٣ آلاف مؤلّف (باعتبار أن كلّ مؤلّف له عشر كتب)، وهو ما سيحتاج إلى ٣٠٠٠ × (٢٠ + ٢) = ٦٦ كيلو بايت.

إذن فإن المجموع الكلّي لمساحة التخزين في حالة تقسيم المعلومات على جدولين = ٦٠ + ٦٦ = ١٢٦ كيلو بايت، أي حوالي خمس المساحة المستهلكة عند وضع أسماء المؤلّفين في نفس جدول الكتب!

هذا بالإضافة إلى سرعة عملية البحث، فالكمبيوتر يكونُ أسرع بكثير عند البحث عن رقم ١٠٠ عنه عند البحث عـن النص "توفيق الحكيم".

ليس هذا فحسب.. افترض أن الذي يدخل أسماء المؤلّفين أخطأ وكتب اسم (توفيق الحكيم) كالتالي: "تفيق الحكيم".. في هذه الحالة ما عليه إلا أن يعدل الخطأ مرة واحدة في جدول المؤلّفين.. أما لو كانت أسماء المؤلّفين في نفـس جـدول الكتب، فإن تكرار كتابة اسم المؤلّف يجعل احتمالات الخطإ أكبر، بالإضافة إلى تضييع الوقت والجهد فـي كتابتهـا، وصعوبة تعديلها كلّها!

طبعا نفس هذا الكلام ينطبق على جدول الناشرين.

قاعدة هامة:

كلّما كان ممكنا، قم بتقسيم البيانات التي تتكرر أكثر من مرة على أكثر من جدول، خاصةً إذا كانـت هـذه البيانـات تستهلك مساحة تخزين كبيرة.. إن هذه العملية تسمى التطبيع Normalization، ولها الكثير من القواعد، وإن كـان من الممكن استنتاجها منطقيا.

قواعد البيانات المترابطة Relational Databases:

رأينا في المثال السابق أن هناك علاقةً Relationship بين جدولي الكتب والمؤلّفين، حيث إن الرابط بينهما هو حقل رقم المؤلّف.. وهنا يجب أن تعرف هذه المصطلحات:

- يسمى الحقل المشترك في كلا الجدولين "حقل مفتاحي" Key field.
- يسمى حقل رقم المؤلّف في جدول المؤلّفين باسم "المفتاح الأساسي" Primary Key، وذلك لأنّه متفرد Unique غير قابل للتكرار.. ولا يحتوي الجدول على أكثر من مفتاح أساسي واحد.
- يسمى حقل رقم المؤلّف في جدول الكتب باسم "المفتاح الفرعي" Foreign Key، لأنّه موضوع في هذا الجدول ليربطه بجدول آخر.. وطبعا يمكن أن تتكرر قيمة هذا العمود بدون أدنى مشكلة.. ويمكن أن يحتوى الجدول على أكثر من مفتاح فرعي تربطه بعدد من الجداول الأخرى.
- يسمى جدول المؤّلفين باسم "الجدول الرئيسي" Master، وجدول أسماء الكتب باسم "جدول التفاصيل" Detail، حيث يمكنك أن تحصل منه على معلومات عن الكتب المتوافرة لكلّ مؤلّف.
- تسمى عملية مطابقة الصفوف بين الجدولين المترابطين عن طريق الحقل المفتاحي، عملية "ربط" Join.. مثل أن تحاول معرفة اسم مؤلّف كتاب معين، أو أن تحاول أن تعرف كلّ الكتب التي كتبها نفس المؤلّف.
- تسمى العلاقة بين الجدولين علاقة "واحد بمتعدد" One-to-many Relationship، وذلك لأن مؤلّفًا واحدا يمكن أن يكون له أكثر من كتاب.
- يمكن أن توجد علاقة "متعدد بمتعدد" Many-to-many Relationship، وذلك لو كان لدينا كتب اشترك في تأليفها أكثر من مؤلّف.. في هذه الحالة سيكون لدينا جدول ثالث هو جدول "مؤلّفو الكتب"، يحتوى على عمود يمثّل رقم الكتاب في جدول عناوين الكتب، وعمود آخر يمثّل رقم المؤلّف في جدول المؤلّفين، كالتالي:

AuthorID	BookID
٢٣	١
١٠٦	١
٨	٢
٨	٣

حيث اشترك المؤلفان رقما ٢٣ و ١٠٦ في تأليف الكتاب رقم ٢، بينما قام المؤلّف رقم ٨ بتأليف الكتابين رقمي ٢ و ٣.. هذه هي علاقة متعدد بمتعدد.

صحة المرجع Referential Integrity:

ماذا سيحدث لو تم حذف سجلّ توفيق الحكيم من جدول المؤلّفين؟.. ستظلّ هناك روايات في جدول الكتب تشير لسجلّ توفيق الحكيم، بينما هو محذوف.. إن هذا وضع مثالي لحدوث أخطاءٍ مزعجة في تطبيقك!

والحلّ؟

يجبُ أن تراعي منطقية وصحة العلاقات المرجعية Referential Integrity بين الجداول، فلو قمت بحذف سـجلّ توفيق الحكيم من جدول الكتب، فيجب أيضا أن يتم حذف كلّ الروايات والكتب التي ألّفها من جدول الكتب.

عامةً اطمئن، فكما ذكرنا من قبل، فإن برامج قواعد البيانات مصممة لتريحك من معظم العمل اليـدوي، لهـذا فهـي ستعينك على التحقّق من صحة العلاقات المرجعية بين الجداول، كما سنرى بعد قليل.

الان سوف نقوم بتجربة لغة ال Sql و لكن قبل ذلك يجب ان نتعرف على برنامج الاكسيس كبرنامج يمكننا تجربة الامثلة بداخلة .

لقد اهتممت بالاكسيس كثيرا هنا لاننا سوف نستخدمة احيانا كثيرة كقاعدة بيانات فى برامجنا لكن لن نستخدم الواجهة المرئية للأكسيس فى برامجنا بل سنستخدم لغة الاستعلام SQL

تشغيل وإنهاء Access 2000:

- تشغيل Access 2000: من قائمة البرامج :-

١. من سطح المكتب انقر على زر Start الموجود في أسفل الشاشة على اليسار في شريط المهام .

٢. تظهر قائمة Start ، نختار منها Programs فتظهر قائمة أخرى نختار منها البرنامج Access Microsoft .

- تشغيل Access 2000 من نافذة My Computer :-

١. من سطح المكتب انقر نقراً مزدوجاً على رمز My Computer .

٢. من نافذة My Computer انقر نقراً مزدوجاً على رمز مشغل القرص C .

٣. انقر نقراً مزدوجاً على مجلد Access 2000 لفتحه .

وبمجرد تشغيل البرنامج يتم فتح مربع حواري يطلب منها إنشاء قاعدة بيانات جديدة باستخدام قاعدة بيانات فارغة أو باستخدام معالجات ، أو فتح قاعدة بيانات موجودة .

Microsoft Access

Create a new database using

Blank Access database

Access database wizards, pages, and projects

Open an existing file

More Files...
Northwind Sample Database
Contacts Sample Database
Address Sample Database
Inventory Sample Database

OK | Cancel

- الشاشة الافتتاحية لبرنامج Access 2000 :-

تتكون شاشة البرنامج من :-

8 شريط القوائم Menu bar : حيث يحتوي على ٧ قوائم وكل قائمة تحتوي على مجموعة من الأوامر .

File Edit View Insert Tools Window Help

ب. شريط الأدوات القياسي Tool bar : يوجد أسفل الشريط السابق ويحتوي على أزار ورموز تستخدم بديلاً للأوامر .



جـ. شريط المعلومات Status bar : يوجد في أسفل الشاشة ويوجد عليه بعض المعلومات عن الملف الفتوح مثل اسمه وعدد سجلاته وموقع المؤشر داخل الملف .

Ready NUM

د. مربع قائمة التحكم Menu Box Control : ويوجد في أقصى اليسار العلوي من الشاشة على شكل مفتاح حيث النقر المزدوج على هذا المفتاح إلى إغلاق البرنامج والنقر الفردي يؤدي إلى إظهار قائمة التحكم .

هـ. زر الإغلاق والتقليص والتكبير : وتوجد في أقصى يمين الشاشة العلوي وتستخدم هذه الأزرار في إلاق النافذة وتصغيرها وتكبيرها .

و. منطقة العمل : وهي المنطقة الكبيرة والتي تظهر بها كائنات قاعدة البيانات التي نتعامل معها مثل الجداول والنماذج والاستعلامات ... الخ

- إنهاء Access 2000 :-

بعد حفظ العمل نقوم بإنهاء البرنامج وذلك بإحدى الطرق التالية :

أ. فتح قائمة ملف ومن القائمة التي تظهر نختار إنهاء .

ب. اضغط مفتاح Alt + F4 .

جـ. انقر نقراً مزدوجاً على مربع قائمة التحكم .

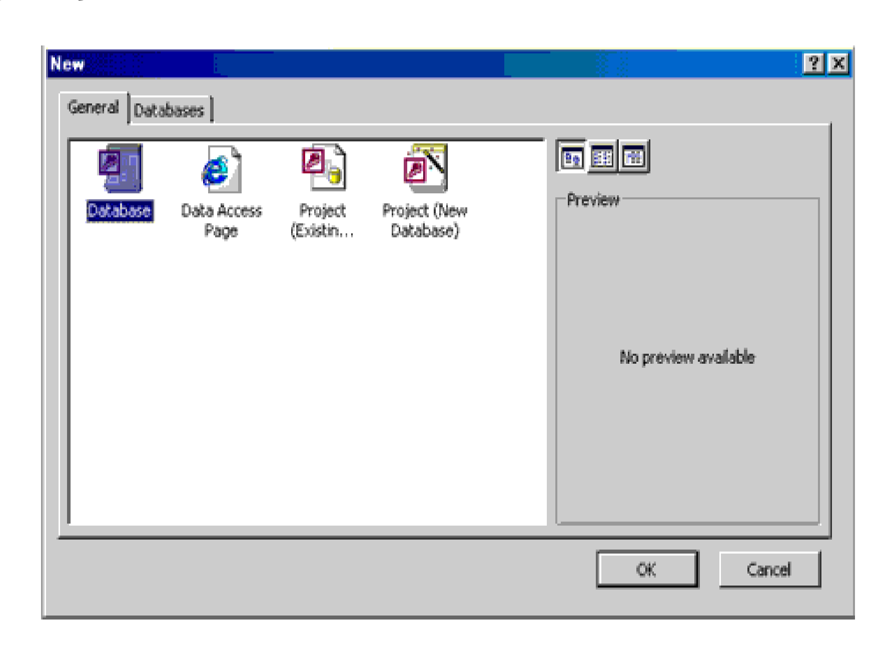
د. انقر زر الإغلاق ✕ .

إنشاء قاعدة بيانات جديدة

- إنشاء قاعدة بيانات فارغة:

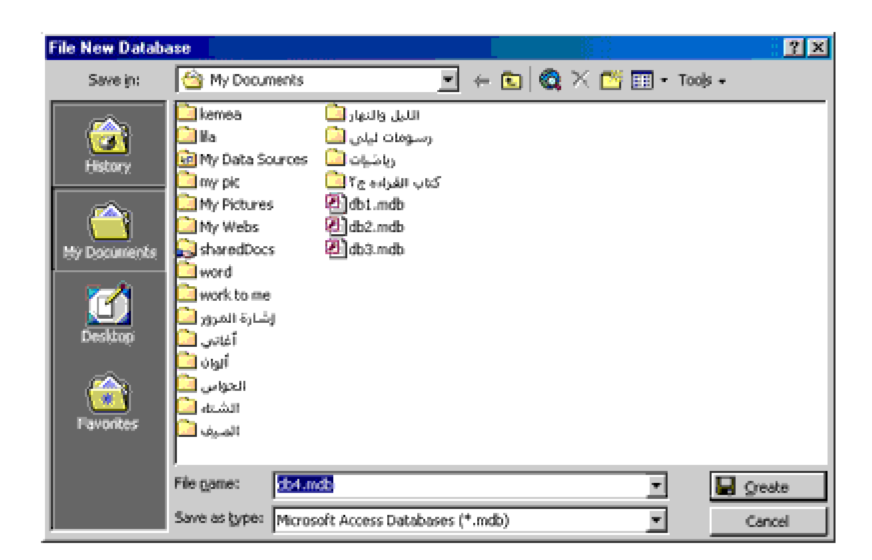
بعد تشغيل البرنامج يتم فتح مربع حوار نختار منه إنشاء قاعدة بيانات فارغة . ثم ننقر على زر 9 موافق.

أما في حالة عدم ظهور مربع الحوار السابق نقوم بفتح قائمة ملف ثم نختار جديد أو ننقر على زر قاعدة بيانات جديدة الموجود على شريط الأدوات ، وفي كلا الحالتين يظهر نافذة (جديد) .



٢. من خلال التبويب General ننقر على قاعدة بيانات ثم نختار موافق فيتم فتح نافذة مربع ملف قاعدة بيانات جديدة .

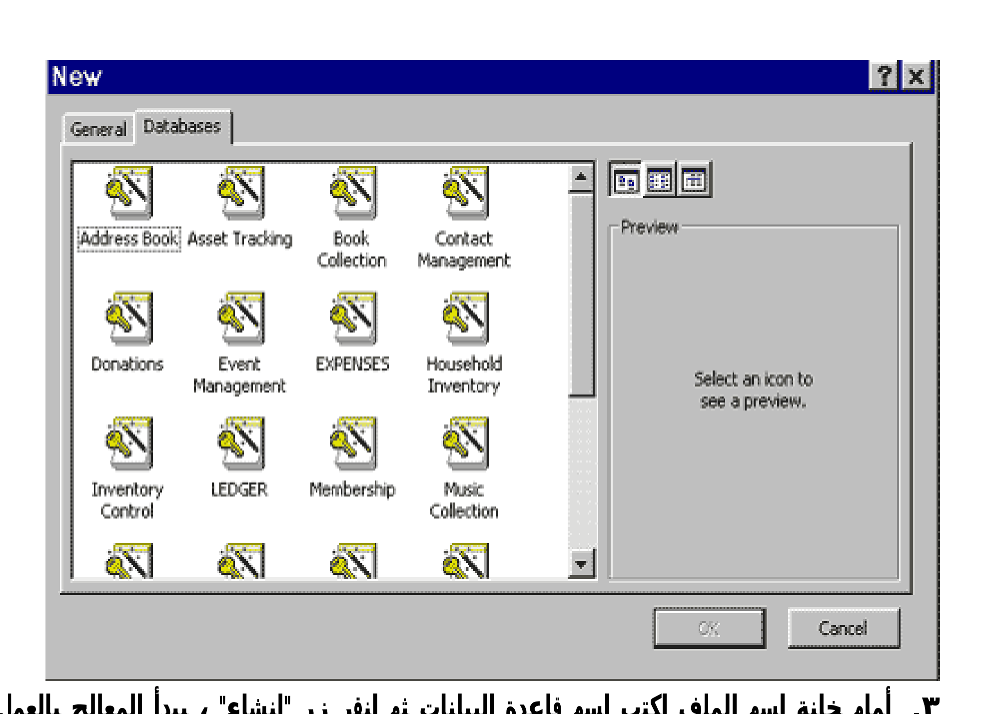
٣. أمام خانة اسم الملف نكتب اسم قاعدة البيانات ثم ننقر على الزر إنشاء ، فيقوم البرنامج بإنشاء قاعدة بيانات جديدة فارغة وبالاسم الذي اخترناه .



10 انشاء قاعدة بيانات باستخدام المعالج :-

١. بعد تشغيل البرنامج يتم فتح مربع حوار نختار منه انشاء قاعدة بيانات باستخدام المعالج ثم ننقر على زر موافق ، أما إذا لم يظهر المعالج السابق فنختار ملف ثم جديد وفي كلا الحالتين يتم اظهار نافذة مربع "جديد" .

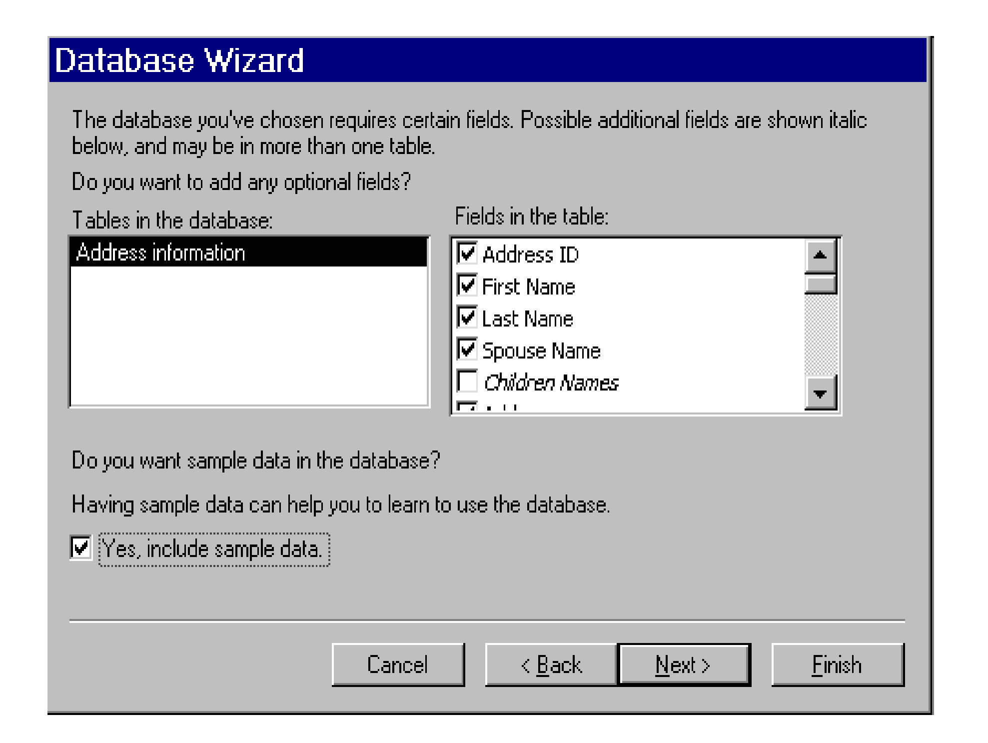
٢. من خلال التبويب Data base انقر أحد المعالجات التي تريدها ثم انقر زر موافق يتم فتح مربع ملف قاعدة بيانات جديدة .



٣. أمام خانة اسم الملف اكتب اسم قاعدة البيانات ثم انقر زر "إنشاء" ، يبدأ المعالج بالعمل وتظهر بعد قليل معلومات توضح ماذا سيفعل المعالج .

انقر زر التالي للاستمرار ، فتظهر بعد قليل قائمة بأسماء الجداول التي أنشأها المعالج لقاعدة البيانات 11 (انظر الشكل) حيث تظهر أسماء الجداول على اليمين بينما تظهر أسماء الحقول الموجودة في الجدول المختار على اليسار .

نقوم باختيار الحقول التي نريدها وذلك بنقر المربع الموجود على يسار الحقل .



٤. انقر زر التالي للاستمرار فيظهر مربع آخر يطلب منك اختيار النمط الذي ترغب باستخدامه في عروض الشاشة ، قم باختيار النمط الذي تريد ثم اضغط على زر التالي ، فيظهر مربع حوار آخر يطلب منك تحديد النمط الذي ترغب باستخدامه في التقارير المطبوعة ، اختر النمط الذي تريده ثم انقر الزر التالي . فيظهر مربع حوار آخر يطلب منا وضع عنوان لقاعدة البيانات وهل نريد تضمين صورة أم لا .

٥. انقر زر التالي فيظهر آخر شكل من مربعات الحوار والذي يسأل هل نريد بدء قاعدة البيانات . نقوم بتنشيط الخيار نعم ثم ننقر على زر إنهاء .

٦. يبدأ المعالج بإنشاء قاعدة بيانات تحتوي على جداول ونماذج وتقارير ... الخ . وبعد الانتهاء من إنشاء قاعدة البيانات تظهر شاشة أخر تحثنا على إدخال البيانات المطلوبة . نقوم بإدخال البيانات وبعد ذلك نغلق النافذة فيظهر إطار آخر اسمه

لوحة التبديل : حيث لا تعتبر هذه اللوحة ذات قيمة كبيرة وانما هي نموذج جميل يسمح لك بأداء الأعمال التي تطلبها من قاعدة البيانات بمجرد النقر على الزر المناسب .

وتظهر هذه اللوحة في كل مرة يتم فتح قاعدة البيانات حيث نقوم بإغلاقها وبمجرد إغلاقها يظهر إطار قاعدة البيانات حيث يحتوي على جميع الجداول والنماذج والاستعلامات ... الخ ، الذي قام المعالج بإنشائها .

- <u>حفظ قاعدة البيانات :-</u>

يمتاز برنامج Access 2000 بميزة حفظ القاعدة بمجرد تسميتها ويتم أيضاً حفظ أي سجل بمجرد إدخاله .
ولكن إذا قمت بعمل أي تغيرات في تصميم جدول أو نموذج أو تقرير وقمت بإغلاقه يظهر مربع حوار يطلب منك هل تريد حفظ التغيرات أم لا .

- إغلاق قاعدة البيانات :-

يتم إغلاق قاعدة البيانات بإحدى الطرق التالية :

١. انقر نقراً مزدوجاً على مربع قائمة التحكم .

٢. انقر زر الإغلاق T .

٣. من قائمة ملف نختار إغلاق .

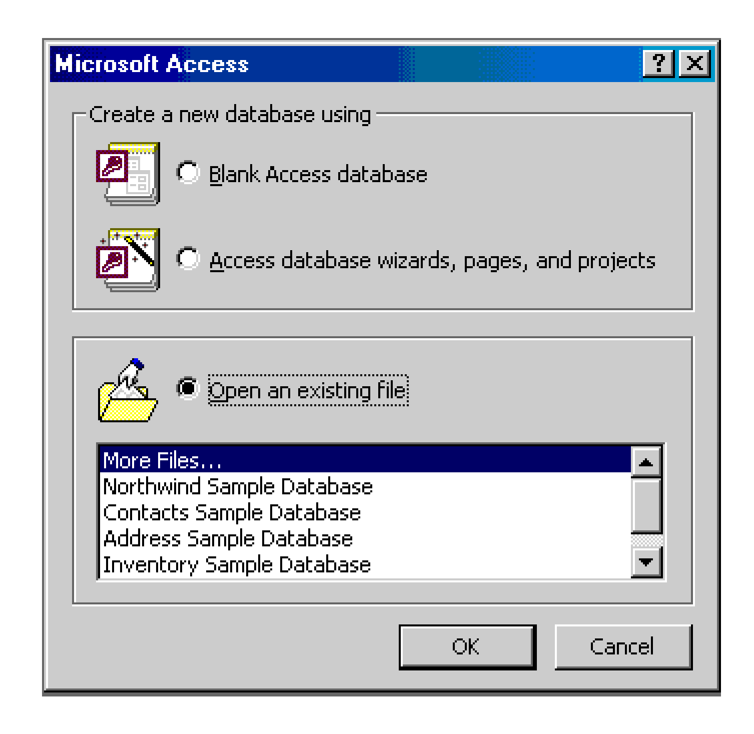
٤. اضغط مفتاح Ctrl +4 F .

- فتح قاعدة البيانات :-

يتم فتح قاعدة البيانات بإحدى الطرق التالية :

١. من قائمة ملف نختار فتح ومن مربع الحوار الذي يظهر نحدد مكان الملف المطلوب ونقوم بفتحه . أو ننقر على زر فتح الموجود على شريط الأدوات .

٢. عند فتح برنامج Access 2000 من مربع الحوار الذي يظهر ، نختار فتح ملف موجود ونقوم بتحديد الملف الذي نريد فتحه

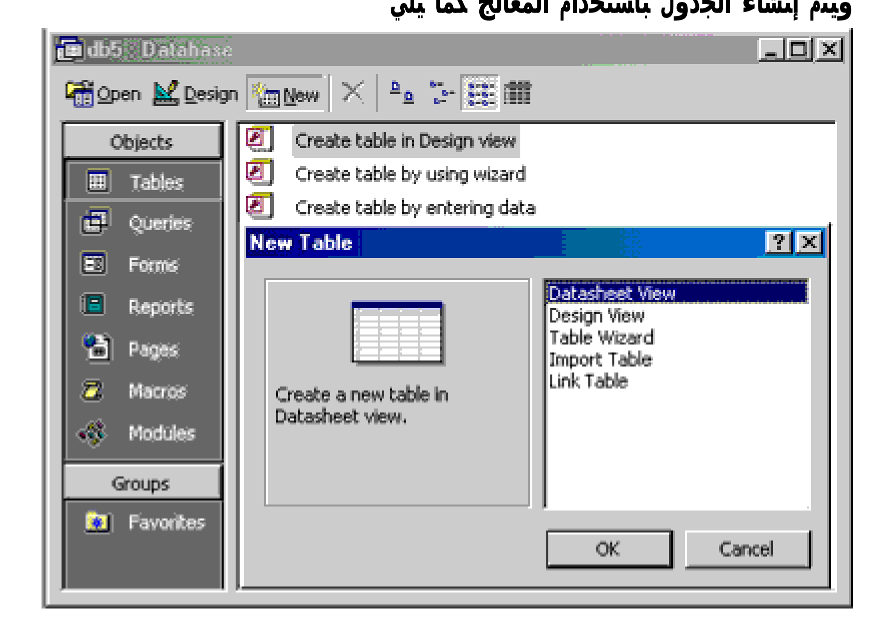


إنشاء الجدول باستخدام المعالج :-

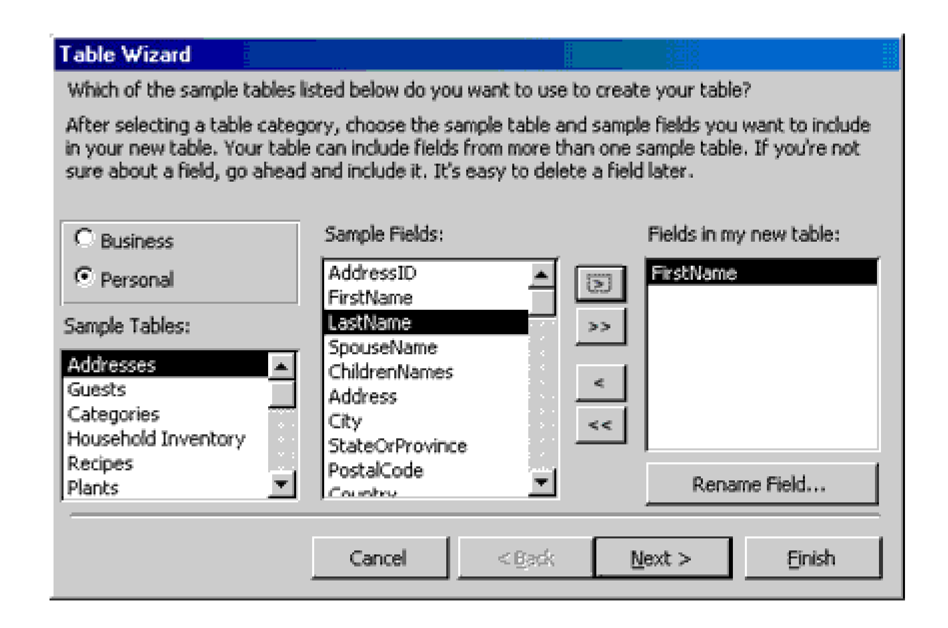
الجدول هو الأساس في أي قاعدة بيانات حيث أن الكائنات الأخرى مثل النماذج والتقارير والاستعلامات تستخرج عادة من بيانات الجداول وليس من أي كائن آخر .

٢. نقوم باختيار التبويب جداول من إطار قاعدة بيانات ثم ننقر على زر "جديد" الموجود في أعلى إطار قاعدة البيانـات ، حيـث يظهر مربع "جدول جديد" (انظر الشكل) ، ويمكن الحصول على نفس المربع من قائمة إدراج أو بالنقر على زر كائن جديـد من شريط الأدوات .

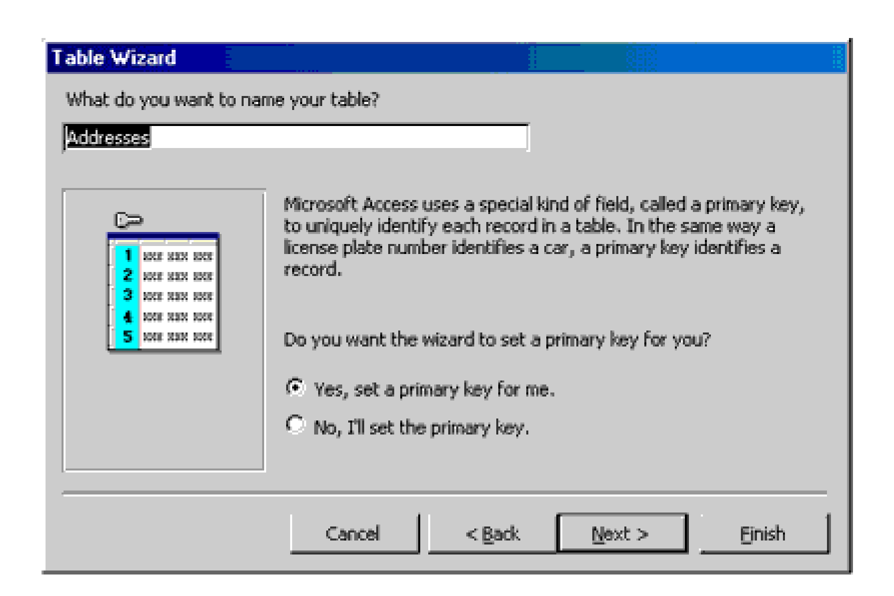
ويتم إنشاء الجدول باستخدام المعالج كما يلي



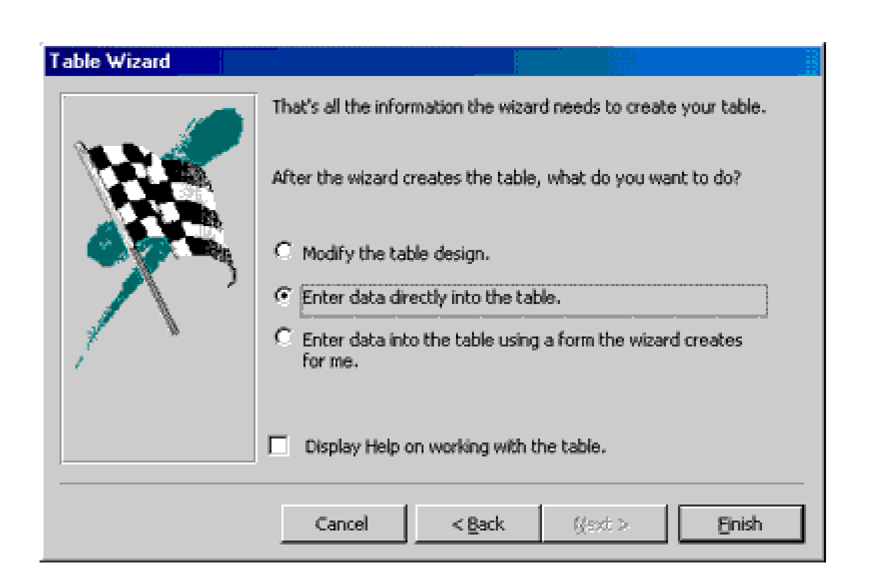
٣. نختار من المربع معالج الجدول ثم ننقر زر موافق فيظهر معالج الجداول (انظر الشكل) ، تظهر تلقائياً قائمة الجداول التي تخص العمل وإذا رغبت في إظهار قائمة الجداول الشخصية نشط خانة الاختيار "شخصي" .



٤. من خانة نماذج الجداول ننقر على الجدول الذي نريده ، حيث تظهر حقول هذا الجدول في خانة نماذج الحقول ، نقوم بعد ذلك باختيار الحقل الذي نريده ثم النقر على زر > حيث يؤدي إلى نقل إلى الجدول الجديد . ومن الممكن أيضاً اختيار حقول أخرى من خانة نماذج الجداول وإضافتها إلى الجدول الجديد ، بعد اختيار الحقول التي نريدها وثم نقلها إلى الجدول الجديد ننقر على زر التالي فيظهر مربع معالج الجداول حيث نقوم بكتابة اسماً للجدول .



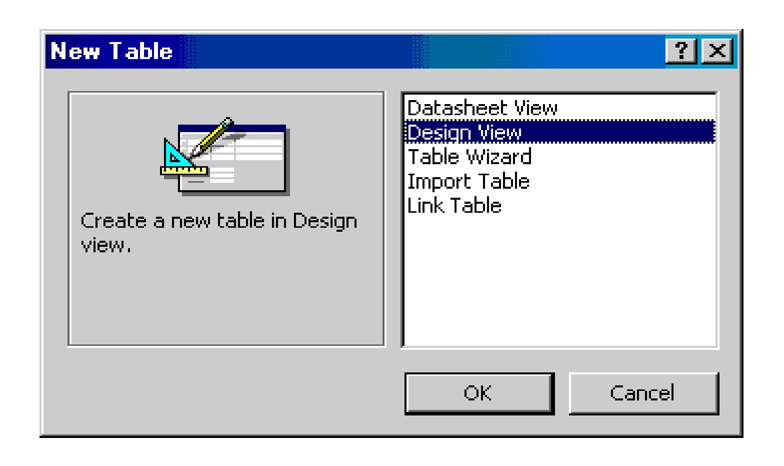
٥. من مربع معالج الجداول نقوم بتنشيط الخيار (نعم) قم بتعيين مفتاح أساسي نيابة عنه ، حيث يقوم المعالج بتعيين حقل معين ليكون مفتاح أساسي ،(المفتاح الأساسي عبارة عن علامة مميزة تكون موجود بجانب كل سجل نقوم بتمييزه عن غيره وذلك لمنع دخول نفس البيانات في نفس الحقل المستخدم كمفتاح أساسي . وعادة يتم اختيار السجل الذي يحتوي رقم وليس على اسم بحيث لا يتكرر). ثم انقر زر التالي فتظهر آخر شاشة من شاشات معالج الجداول ، حيث يحتوي على خيارات كثيرة .



٦. انقر على الخيار الذي تريده ثم انقر زر إنهاء فيتم إنشاء الجدول حسب الاختيار الذي تم تحديده بالسابق وبعد عمل المطلـوب قم بإغلاق الجدول .

إنشاء جدول بدون المعالج :-

١. من إطار قاعدة البيانات نشط التبويب جداول ثم انقر الزر جديد فيظهر مربع (جدول جديد) ومن هذا المربع انقر "طريقة عرض التصميم" ثم انقر زر موافق فتظهر نافذة عرض تصميم الجدول .



٢. في أول سطر وتحت عمود اسم الحقل اكتب اسم أول حقل بعد ذلك انقل المؤشر إلى العمود الثاني (عمود نوع البيانات) حيث يوجد بجانبه سهم صغير وعند الضغط عليه تظهر قائمة مسدلة له تحتوي على مجموعة من أنواع الحقول مثل (نص ، رقم ، تاريخ، نعم / لا ، عمله ، مذكر ، ترقيم تلقائي ، كائن ، معالج البحث) قم باختيار نوع الحقل حسب العمود الذي قبله اسم الحقل .

٣. انتقل إلى العمود الثالث (الوصف) ثم اكتب وصف للحقل الذي تعمل عليه ، وهو أمر اختياري .

Table1 : Table

Field Name	Data Type	Description

Field Properties

General | Lookup

The data type determines the kind of values that users can store in the field. Press F1 for help on data types.

٤. نلاحظ أيضاً في النصف السفلي من نافذة عرض التصميم تظهر خصائص الحقل الحالي المختار قم بتحديد خصائص الحقل بما يناسبك . مثل (الحجم ، التنسيق ، الأماكن العشري ، قناع الإدخال ، تعليق ... الخ) .

٥. قم بتعبئة أسماء جميع الحقول التي ترغب بها مع تحديد نوع البيانات والخصائص .

٦. انقر زر الإغلاق فتظهر رسال تخبرك بحفظ التغيرات ، اختر نعم ، فيظهر مربع حفظ باسم اكتب اسماً للجدول ثم انقر موافق .

ضبط المفتاح الأساسي بدون معالج :-

يجب تخصيص حقل أو أكثر من حقول الجدول وجعله مفتاح أساسي Primary Key ويراعى عند اختيار الحقل أن يشتمل على بيانات لا يسمح بتكرارها داخل الجدول مثل رقم حساب العميل أو الرقم الوطني ... الخ .

ولعمل ذلك :

١. ضع المؤشر أمام الحقل حتى يتحول إلى سهم أسود صغير ثم انقر زر الفأرة حيث يظهر رأس سهم صغير على يمين اسم الحقل ويتم إضاءة السجل كله .

٢. وجه المؤشر إلى شريط الأدوات واضغط على زر المفتاح ، فيظهر رمز المتفاح على يمين اسم الحقل دلالة على أن هذا الحقل أصبح مخصصاً كمفتاح أساسي .

Table1 : Table

Field Name	Data Type	Description
school arabia	Text	

Field Properties

General | Lookup

Field Size	50
Format	
Input Mask	
Caption	
Default Value	
Validation Rule	
Validation Text	
Required	No
Allow Zero Length	No
Indexed	Yes (No Duplicates)
Unicode Compression	Yes

Table1 : Table

Field Name	Data Type	Description
school arabia	Text	

Field Properties

General | Lookup

Field Size	50
Format	
Input Mask	
Caption	
Default Value	
Validation Rule	
Validation Text	
Required	No
Allow Zero Length	No
Indexed	Yes (No Duplicates)
Unicode Compression	Yes

التبديل بين عرض التصميم وعرض صحفة البيانات :-

يسمح برنامج Access 2000 بعرض جداول البيانات بطريقتين :

١. طريقة عرض التصميم : حيث يظهر فيها أسماء الحقول وخصائصها .

٢. طريقة عرض صفحة البيانات : حيث يظهر فيها البيانات المسجلة بالجدول .

ويتم التبديل بين طريقتي العرض عن طريق الضغط على زر (عرض الجدول) الموجود في أقصى اليسار من شريط الأدوات . واختيار طريقة العرض . أو عن طريق فتح قائمة عرض ثم اختر طريقة العرض .

س١: يوجد أنواع متعددة من البيانات فكيف اختار من بينها النوع الذي يناسبني ؟ أرجو أن تعطيني توضيح عنها .

ج: أنواع البيانات كالتالي :

١- نص : ويمكنه تخزين حروف أو أرقام أو كلاهما وأقصى حجم له ٢٥٥ حرف .

٢- مذكرة : حروف أو أرقام أوكلاهما وأقصى حجم له ٦٥،٥٣٥ حرف مما يجعله ينفع في البيانات النصية التي يتجاوز عدد حروفها ٢٥٥ حرف .

٣- رقم : يتم تخزين البيانات كأرقام وحجمها يختلف حسب نوع الرقم .

٤- تاريخ/وقت : بيانات الرقم والتاريخ .

٥- عملة : بينات رقمية تصل دقتها حتى ١٥ رقماً إلى يسار فاصل العلامة العشرية و ٤ أرقام إلى اليمين.

٦- ترقيم تلقائي : حقل رقمي يتزايد بمعدل واحد لكل سجل أو يقوم أكسس باختيار رقم عشوائي .

٧- نعم/لا : تستخدم عند كون البيانات مكونة من قيمتين إما "نعم" أو"لا" .

٨- كائن OLE : كائن مثل جدول بيانات Microsoft Excel أو مستند Microsoft Word أو رسومات أو أصوات وتكون هذه البيانات إما مرتبطة بالكائن الأصلي أو غير مرتبطة.

٩- ارتباط تشعبي : نص أو تركيبات من نص وأرقام يتم تخزينها كنص وتستخدم كـ عنوان ارتباط تشعبي. يضم عنوان الارتباط التـــــــــــــــشعبي حتـــــــــــــــى ثلاثـــــــــــــــــــة أجـــــــــــــــزاء:

نـــــص للعـــــرض : الـــــنص الـــــذي يظهـــــر فـــــي حقـــــل أو عنـــــصر تحكـــــم.

عنـــــــــــوان : المـــــــــــسار إلـــــــــــى ملـــــــــــف أو الـــــــــــصفحة .

عنـــــــوان فرعـــــــي : موقـــــــع فـــــــي الملـــــــف أو الـــــــصفحة.

تلميح شاشة : النص المعروض كتلميح أدوات التحكم .

أسهل طريقة لإدراج عنوان ارتباط تشعبي في حقل أو عنصر تحكم هي النقر فوق ارتباط تشعبي في القائمة إدراج.

معالج البحث : إنشاء حقل يسمح لك باختيار قيمة من جدول آخر أو من قائمة بالقيم باستخدام مربع 7 **نص أو مربع تحرير وسرد. يؤدي النقر فوق هذا الخيار إلى بدء تشغيل "معالج البحث" الذي يقوم بتعيين نوع البيانات استناداً إلى Microsoft Access بإنشاء حقل بحث. بعد إكمال المعالج، يقوم القيم المحددة في المعالج.**

تعديل الجداول

قبل القيام بعملية تعديل الجدول يجب أن يتم فتح الجدول بطريقة عرض التصميم .

تعديل الحقول وخصائصها :-

١. بعد عرض الجدول بطريقة عرض التصميم من عمود اسم انقر اسم أي حقل لاختيار ، وقم بتعديل ما تريد .
٢. من عمود نوع البيانات قم بتعديل نوع بيانات الحقل .
٣. في مربع خصائص الحقل الذي يظهر بالأسفل قم بتغيير الخاصية التي تريدها سواءً بنقر مربع الكتابة الذي يظهر أمام الخاصية أو من خلال القوائم المنسدلة .
٤. كرر الخطوات السابقة لكل حقل ترغب في تعديل اسمه أو نوع بياناته أو خصائصه .

إضافة حقول جديدة :-

١. اختر الحقل الذي ترغب في إضافة حقل جديد قبله .
٢. من شريط الأدوات انقر زر إدراج صفوف حيث يظهر صفاً خالياً من البيانات .
٣. اكتب اسم الحقل ونوع البيانات .

حذف حقول من الجدول :-

١. اختر الحقل الذي ترغب في حذفه .
٢. من شريط الأدوات انقر زر حذف صفوف ، أو اضغط مفتاح Del ولحذف الحقل في طريقة عرض صفحة البيانات :
أ) اختر العمود الخاص بالحقل الذي ترغب في حذفه .
ب) افتح قائمة تحرير ثم اختر الأمر حذف عمود .
جـ) تظهر رسالة تحذيرية اختر نعم لتأكيد الحذف .

إدخال سجل إلى جدول :-

١. افتح الجدول في طريقة عرض صفحة البيانات .

٢. بمجرد كتابة آخر سجل في الجدول يتم فتح سجل جديداً تحته انتظاراً لكتابة سجل آخر ، ويتم حفظ السجل بمجرد الانتقال إلى سجل جديد . وأثناء إضافة السجلات تظهر رموز على يمين السجل وهذه الرموز هي :-

▶	يعني هذا الرمز أن هذا السجل هو الحالي .
✎	يعني هذا الرمز أن هذا هو المكان الذي سيدخل فيه سجلاً جديداً
✳	يعني هذا الرمز أن تغيير حدث على السجل ولكنه لم يحفظ بعد

اختيار السجلات :-

١. لاختيار سجل بالكامل وجه المؤشر إلى يمين السجل وعندما يتحول المؤشر إلى سهم ، انقر زر الفأرة الأيسر .

٢. لاختيار سجلات متجاورة اختر أول سجل ثم اضغط مفتاح Shift واستمر ضاغطاً أثناء اختيار باقي السجلات ، أو استخدم الفأرة باختيار السجل الأول ثم السحب .

٣. لاختيار كل السجلات افتح قائمة تحرير ثم اختار تحديد كافة السجلات .

حذف السجلات :-

١. اختر السجل أو السجلات المطلوبة .

٢. اضغط مفتاح Del .

نقل ونسخ البيانات :-

١. اختر البيانات التي تريد نسخها سواءً كانت خلية أو سجل أو مجموعة سجلات .

٢. ثم اختر الامر نسخ من شريط الأدوات .

٣. حدد المكان الذي سوف تنسخ إليه البيانات .

٤. اختر الأمر لصق من شريط الأدوات .

الانتقال داخل الجدول :-

١. يمكن استخدام الفأرة لاختيار أي حقل أو سجل .

٢. يمكن استخدام لوحة المفاتيح للتنقل داخل الجدول .

تغيير عرض الأعمدة :-

١. وجه المؤشر إلى الخط الرأس الذي يفصل بين أسماء الحقول وعندما يتحول المؤشر إلى شكل سهم برأسين ١ اسحب الخط الرأسي لجهة اليسار أو اليمين لزيادة عرض العمود .
٢. أو اختر العمود أو الأعمدة التي نريد تغيير عرضها ثم افتح قائمة تنسيق ثم اختر أمر "عرض العمود" فيظهر مربع (عرض العمود) .

قم بضبط عرض العمود وذلك بكتابة رقم عرض العمود داخل خانة عرض العمود . أو قم باختيار الاحتواء الأفضل وذلك لضبط حجم العمود ليتناسب تماماً مع البيانات الموجودة به .
بعد ذلك انقر زر موافق .

تغيير ارتفاع الصفوف :-

وجه المؤشر إلى عمود اختيار السجل ثم ثبته على أي خط من الخطوط الشبكية التي تظهر تحت السطور ، فيتغير شكل المؤشر إلى سهم برأسين ٢ ، اسحب السهم لأسفل لتزيد من ارتفاع السطور .

تغيير خط الكتابة :-

إن اختيار الخط الذي نريد سوف يؤثر على كل بيانات الجدول ولن يؤثر على الخط الموجود في النماذج والتقارير .
لاختيار خط اتبع ما يلي :-
افتح قائمة تنسيق ثم اختر أمر خط فينظهر مربع حوار خط . 8

Arial (Arabic) 8 B I U

٢. من خانة الخط حدد نوع الخط المطلوب .
٣. من خانة النمط حدد النمط الذي تريده .
٤. من خانة الحجم حدد الحجم الذي تريده .
٥. من خانة اللون حدد اللون الذي تريده .
٦. انقر زر موافق .

تجميد الأعمدة وإعادة تحريرها :-

تستخدم فكرة تجميد الأعمدة لتثبيت حقل معين أثناء طي الشاشة لرؤية الحقول الأخيرة من الشاشة مع الحقل الأول مثلاً . ولعمل ذلك :-

١. اختر العمود الذي تريد تجميده .

٢. افتح قائمة تنسيق ثم اختر تجميد أعمدة من القائمة المنسدلة .

٣. انقر شريط التمرير الأفقي إلى الجهة المعاكسة للعمود الذي اخترت تجميده . سوف ترى بأن العمود الذي اختر تجميده سوف يبقى ثابتاً وبقية الأعمدة سوف تتحرك .

٤. لإزالة التجميد ، افتح قائمة تنسيق ثم اختر تحرير كافة الأعمدة من القائمة المنسدلة

البحث عن البيانات وترتيبها

البحث عن المعلومات يعني توجيه سؤال والحصول على الإجابة . وتوجد عدة طرق للبحث منها :

البحث باستخدام خاصية البحث :-

للبحث عن سجل معين ابتع الخطوات التالية :

١. من صفحة البيانات اختار العمود الذي يحتوي على المعلومة التي تبحث عنها .

٢. من شريط الأدوات انقر زر بحث فيظهر مربع حواري بحث واستبدال .

Find and Replace
Find
Replace
Find what:
Search options
Find where:
All pages
Selected page(s)
Current Page
Direction:
All
Match case
Find whole word only
Match diacritics
Match kashida
Match alef hamza
Find Next
Cancel

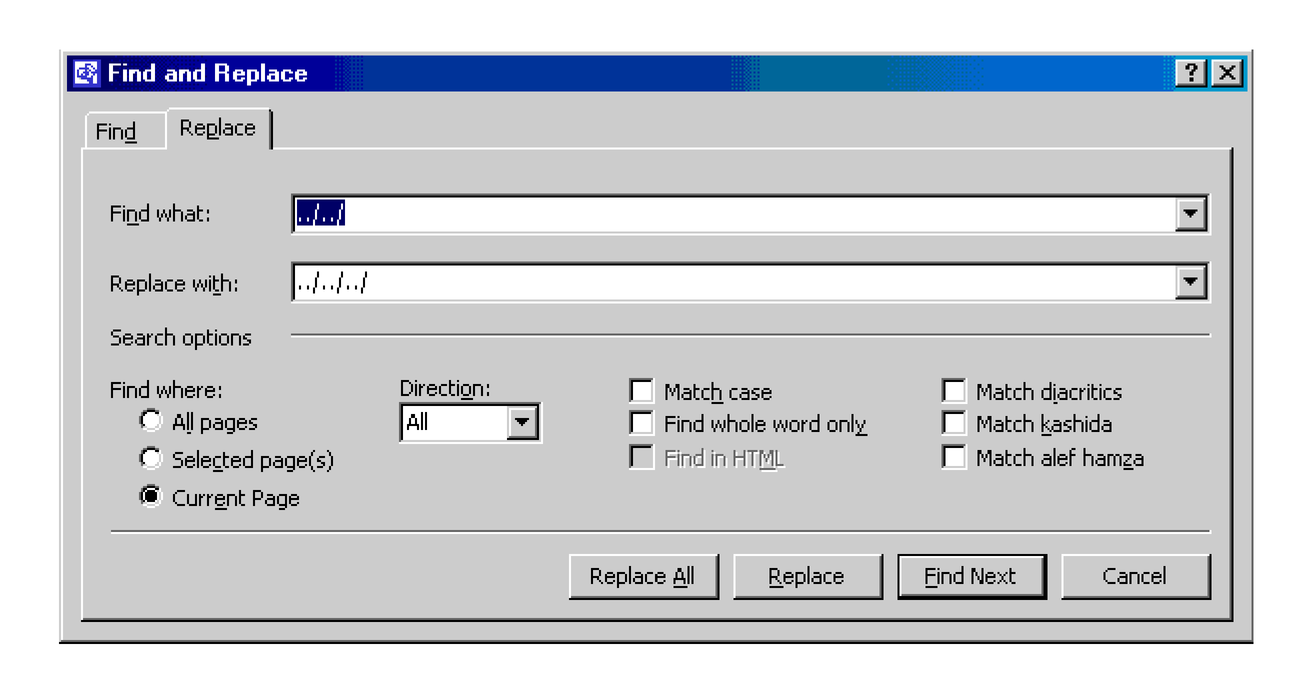
٣. اختر التبويب بحث ثم اكتب في خانة البحث عن النص الذي تبحث عنه .
وفي خانة البحث في حدد اسم الحقل الذي تريد البحث فيه ثم انقر على زر بحث عن التالي . فيتم في هذه الحالة البحث عن السجل المطلوب وعندما يجده يضعه تحت الشريط المضاء .
انقر زر الإغلاق لتعود إلى جدول البيانات وقراءة البيانات التي تبحث عنها . 9

البحث بجزء من المعلومة :-

نستخدم للبحث عن سجل لا نعرف إلا جزء من النص .
١. من مربع الحوار السابق انقر الزر ، الموجود في خانة مطابقة فتظهر قائمة منسدلة بالاختيارات التي يمكن البحث فيها . (انظر الشكل)
٢. اختر (أي جزء من الحقل) ثم انقر زر بحث عن التالي فيقوم البرنامج بالبحث عن أول سجل توجد به المعلومة المتوفرة وتضعه تحت الشريط المضاء .

البحث باستخدام خاصية الاستبدال :-

نستخدم هذه الخاصية للبحث عن معلومة معينة واستبدالها بواحدة أخرى .
اختر العمود الذي يحتوي على المعلومة التي تريد استبدالها ، ثم اختر أمر استبدال من قائمة تحرير ، 10
فيظهر مربع الحوار السابق ، قم بتنشيط التبويب استبدال .



٢. اكتب المعلومة التي تبحث عنها في خانة البحث عن .
٣. اكتب المعلومة التي تريد استبدال القديمة بها في خانة استبدال بـ .
٤. حدد الحقل الذي تريد البحث فيه .

٥. انقر زر بحث عن التالي فيقوم البرنامج بالبحث عن المعلومة وعندما يجدها يضعها تحت الشريط المضاء .
٦. انقر زر استبدال وذلك لاستبدال المعلومة القديمة بالجديدة .

استخدام خاصية التصفية Filtering للبحث عن مجموعة سجلات :-

يستخدم عامل التصفية لعزل سجلات تحتوي على معلومة معينة ، وتوجد عدة طرق لفرز السجلات منها :-

١. التصفية حسب التحديد :-

أ) افتح الجدول الذي تريد ، ثم حدد الحقل الذي تريد تصفية السجلات تبعاً لمحتويات ثم حدد القيمة التي تريد التصفية على أساسها .

ب) انقر زر تصفية حسب التحديد من شريط الأدوات فينتج فرز البيانات حسب التحديد الذي حددته .

جـ) بعد الاطلاع على السجلات انقر زر إزالة عامل التصفية .

٢. التصفية حسب النموذج :-

يتم في هذه التصفية استخدام أكثر من معيار لتصفية السجلات .

أ) افتح الجدول الذي تريد عمل تصفية له ثم انقر زر "تصفية حسب النموذج" من شريط الأدوات .
يظهر نموذج خال بعنوان "تصفية حسب النموذج" يحتوي على سجل واحد بدون بيانات .

ب) وجه المؤشر إلى الحقل الذي تريد ثم انقر السهم المتجه إلى اسفل في داخل الخلية واختر المعيار الذي تريده من القائمة المنسدلة .

جـ) وجه المؤشر إلى حقل آخر وانقر السهم المتجه إلى أسفل داخل الخلية واختر المعيار الثاني الذي تريده وهكذا .

د) انقر التبويب (أو) الذي يظهر في أسفل المربع الحواري فيظهر سطر خال من البيانات لنكتب الشرط الثاني وهكذا يمكن إضافة العديد من الشروط لمعايير التصفية .

هـ) بعد الانتهاء من كتابة كل معايير التصفية انقر زر (تطبيق عامل التصفية) من شريط الأدوات فتظهر السجلات التي ينطبق عليها معايير التصفية الذي حددتها .

و. بعد الانتهاء انقر على زر إزالة عامل التصفية / الفرز .

فرز السجلات تصاعدياً :-

لترتيب سجلات الجدول ترتيباً تصاعدياً أي بحسب الحروف الأبجدية من الألف إلى الياء أوحسب الارقام من صفر إلى ٩ نقوم بما يلي :-

١. انقر أي سجل من سجلات الجدول في الحقل الذي سيتم الفرز طبقاً لمحتوياته .

٢. انقر زر (فرز تصاعدي) من شريط الأدوات .

فرز السجلات تنازلياً :-

أي الفرز حسب الحروف الأبجدية من الياء إلى الألف أو حسب الأرقام من ٩ إلى الصفر :-

١. انقر أي سجل من سجلات الجدول في الحقل الذي سيتم الفرز طبقاً لمحتوياته .

٢. انقر زر (فرز تنازلي) من شريط الأدوات .

إنشاء الاستعلامات واستخدامها Queries

الاستعلام هو تطبيق معايير بحث على بيانات الجدول ثم استعراض سجلات البيانات التي تتطابق مع الشروط المحددة .

طرق إنشاء الاستعلام : توجد طريقتين لإنشاء الاستعلام هي باستخدام المعالج أو بدون استخدام المعالج (بنفسك) (طريقة عرض التصميم)

١. إنشاء استعلام باستخدام المعالج :

يحتوي البرنامج على أربعة أنواع من المعالجات المستخدمة في الاستعلام هي :

أ) معالج الاستعلامات البسيط : وهو من أسهل أنواع الاستعلامات ، وهو أكثر أنواع الاستعلامات استخداماً ، حيث لا يتضمن أي معايير أو شروط يمكن تطبيقها على سجلات الجدول / الجداول .

ب) معالج الاستعلامات الجدولية : حيث يظهر ملخصات مثل المجموع والعدد والمتوسط الحسابي لبيانات حقل معين ، ويضعهم في مجموعة واحدة .

جـ) معالج استعلام البحث من التكرار : حيث يقارن بين جدولين ويبحث عن السجلات المتطابقة في كليهما .

د) معالج استعلام البحث من غير المتطابقات : حيث يقارن بين جدولين ويبحث عن السجلات غير المتطابقة في كليهما .

إنشاء استعلام باستخدام معالج الاستعلامات البسيط :

أ) افتح ملف قاعدة بيانات ، ثم نشط التبويب استعلامات .
ب) انقر الزر جديد فيظهر مربع استعلام جديد .

New Query

Create a new query without using a wizard.

Design View
Simple Query Wizard
Crosstab Query Wizard
Find Duplicates Query Wizard
Find Unmatched Query Wizard

OK Cancel

جـ) انقر (معالج الاستعلامات البسيط) ثم انقر زر موافق فيظهر أول مربع من مربعات معالج الاستعلامات البسيطة . (انظر الشكل) ، ممكن أداء الخطوتين السابقتين بخطوة واحدة وذلك عن طريق نقر إنشاء استعلام باستخدام المعالج من إطار قاعدة البيانات نقراً مزدوجاً

د) اختر الجدول الذي ستختار منه حقول الاستعلامات وذلك من خانة (جداول / استعلامات)

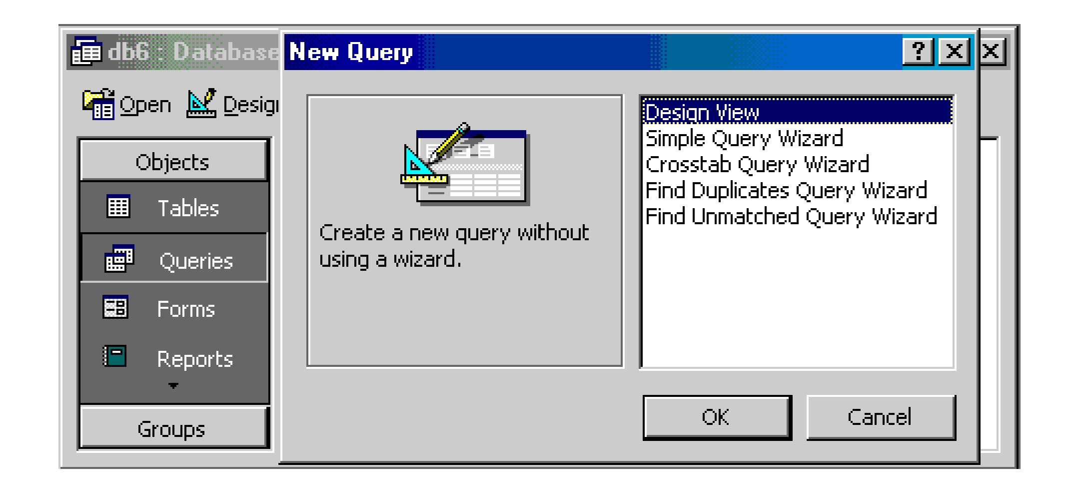
هـ) من خانة (الحقول المتاحة) حدد الحقول التي تريدها ثم قم بنقلها إلى خانة (الحقول المحددة) عن طريق نقر الزر > بعد ذلك انقر زر التالي . فتظهر نافذة أخرى تطلب منك تحديد اسم للاستعلام اكتب الاسم الذي تريده ثم انقر على زر إنهاء .تظهر بعد ذلك نتيجة الاستعلام في طريقة عرض صفحة بيانات .

***وممكن ايضاً عرضه بطريقة عرض التصميم وذلك للتعديل عليه .**

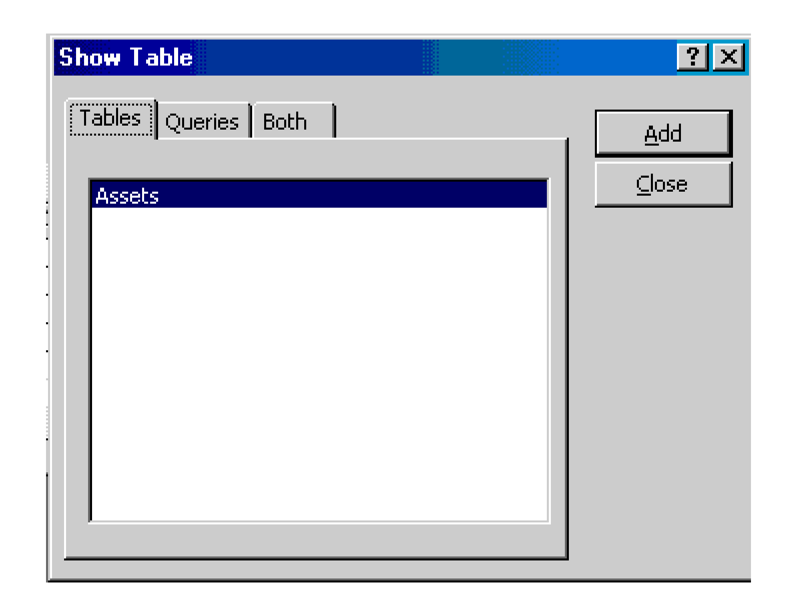
<u>٢. انشاء استعلام بطريقة عرض التصميم (بنفسك) :</u>

أ) افتح ملف قاعدة بيانات ثم نشط التبويب استعلامات .

ب) انقر زر جديد فيظهر مربع استعلام جديد



جـ) انقر (طريقة عرض التصميم) ثم انقر زر موافق ، فيظهر مربع (إظهار جدول)



يشتمل هذا المربع على ٣ تبويبات وهي :

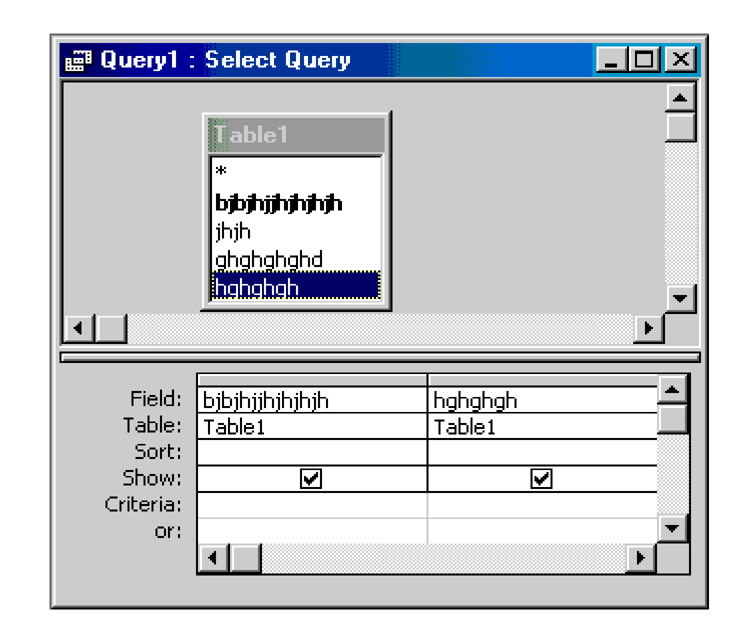
١. جداول : يظهر قائمة بأسماء الجداول الموجودة .

٢. استعلامات : يظهر قائمة بأسماء الاستعلامات الموجودة .

٣. كلاهما : يظهر قائمة بأسماء الجداول والاستعلامات الموجودة .

د) نشط التبويب جداول ثم انقر نقراً مزدوجاً على الجداول المطلوب أو حدد الجدول ثم انقر زر إضافة (إذا قمت بإضافة أكثر من جدول يجب أن تنشأ علاقة ارتباط بينهما) .

هـ) قم بإغلاق مربع (إظهار جدول) فيظهر مربع (استعلام تحديد)



حيث يحتوي هذا المربع على قسمين علوي وسفلي ، القسم العلوي يحتوي على الجدول / الجداول الذي تم اختياره بالسابق والقسم الآخر يحتوي على منطقة معايير الاستعلام (شبكة QBE) .

و) قم بسحب الحقول التي تريدها من الجدول إلى منطقة معايير الاستعلام .

ي) انقر زر (عرض صفحة البيانات) الموجود على شريط الأدوات فتظهر شاشة تحتوي على البيانات المختارة .

ملاحظة : تتم عمل التعديلات المطلوبة سواءً حذف أو إضافة أو ترتيب أو فرز بطريقة عرض التصميم .كما يمكن إضافة معايير معينة إلى الاستعلام وذلك في طريقة عرض التصميم أمام سطر معايير من شبكة QBE . وذلك لتحديد شرط أو أكثر من شرط

لغة SQL

لا بد أنك قد لاحظت الآن أن هنالك الكثير من العمليات التي تستطيع قاعدة البيانات أدائها، فيمكنها أن تنشأ سجلا أو تحذف سجلات أو تعدل سجلا أو تغير جدولا أو تحذف جدولا وغيرها الكثير، هذه العمليات تجرى بواسطة لغة SQL وهي اختصار لعبارة Structured Query Language أي لغة الاستعلامات البنيوية، تسمى كل عملية تنفذ عن طريق لغة SQL بالاستعلام Query، وعلى حسب لغة البرمجة التي تستخدمها وبرنامج قواعد البيانات الذي تستخدمه فإن الاستعلامات ترسل وتنفذ بطرق مختلفة وكيفية الحصول على نتائج الاستعلام تختلف أيضا، في هذه الدورة لن يهمنا كيف تقوم بإرسال الإستعلام إلى قاعدة البيانات وتنفذه وتحصل على نتائجه، ولكن ما سنتحدث عنه هو كيفية كتابة الاستعلام نفسه، لغة SQL شبه متفق عليها بين جميع نظم قواعد البيانات، على سبيل المثال في موقعنا نستخدم لغة البرمجة PHP للوصول إلى مزود قاعدة البيانات MySQL، وهي من التوليفات الشهيرة، ومنها أيضا استخدام برنامج Visual Basic للوصول إلى مزود قاعدة بيانات Microsoft SQL Server لعمل البرامج التي تتطلب وصولا إلى بيانات مركزية كما في المحلات والأسواق والمخازن والمستشفيات وغيرها، ويمكن استخدام لغة SQL أيضا عن طريق البرامج المكتوبة بلغة C و C++ و Perl و Delphi للوصول إلى بيانات مخزنة في قواعد بيانات Oracle و Sybase و Informix و IBM/DB2 و Access وغيرها.

مزودات قواعد البيانات

من الطرق الشهيرة لتخزين قواعد البيانات هي وضعها في صورة ملفات نصية بسيطة Plain Text، بحيث يوضع كل سجل في سطر من أسطر الملف ويفصل بين الحقول المختلفة في كل سجل بفاصلة comma (،)، وتكون أحيانا في صورة ملفات أكثر تعقيدا بحيث يحتوي الملف على الكثير من الجدوال والفهارس التي تسرع عمليات البحث في قواعد البيانات والاستعلامات الجاهزة والنماذج كما في قواعد بيانات Microsoft Access.

هذه الطريقة قد تعتبر جيدة في البرامج البسيطة ولكن في بيئات الشبكات المعقدة والبرامج التي تحتاج وصولا مشتركا إلى البيانات تصبح هذه الفكرة صعبة التطبيق حيث أن ملف البيانات يكون في جهاز المزود، ويتم الوصول إليه عبر الشبكة من قبل الزبون، وفي حالة الملفات الثابته يجب أن يتم نقل ملف البيانات بأكمله عبر الشبكة حتى يصل إلى الزبون الذي يأخذ المعلومات منه وإذا قام الزبون بتغيير أي من المعلومات فإن المزود يجب أن ينتظر حتى يقوم الزبون بإعادة الملف بعد عمل التغييرات عليه، وهذا أمر يستغرق الكثير من الوقت خاصة إذا كان حجم الملف كبيرا وكانت الشبكة بطيئة، وكذلك إذا أراد أكثر من شخص الوصول إلى البيانات في نفس الوقت وتغييرها في نفس الوقت سيؤدي ذلك إلى عطب البيانات وحدوث خلل فيها، لهذا فإن المزود يقوم بإعطاء الملف لمن يطلبه أولا ويقوم

بقفل الملف بحيث يجب على كل من يطلب الملف بعد ذلك أن ينتظهر حتى ينتهي هذا الشخص من الملف، وبعدها الذي يليه وهكذا، وهو أمر غير معقول أبدا!

الحل لهذه المشكلة كان بعمل ما يسمى بمزود SQL، يقوم مزود SQL باستقبال جميع الأوامر والطلبات في صورة استعلامات SQL ثم يقوم بتنفيذ هذه العمليات على المزود ويرسل نتيجة التنفيذ فقط إلى الزبون دون الحاجة إلى إرسال قاعدة البيانات بأكملها، فإذا أردت أن تحصل على حقل واحد فقط من أحد السجلات في أحد الجداول في قاعدة البيانات فإن الجزء الذي سيتم إرساله عبر الشبكة هو الطلب في صورة SQL والجزء الذي سيتم إعادته عبر الشبكة هو هذا الحقل فقط والذي قد لا يصل في حجمه إلى ١٠ بايتات مثلا إذا كان مكونه من ١٠ أحرف، مقارنة مع عملية إرسال قاعدة البيانات بأكملها والتي قد يصل حجمها إلى العديد من الميجابايتات وربما الجيجابايتات في بعض الأحيان، إضافة إلى ذلك فإن مزود SQL يقوم بتوزيع الأعباء على الطلبات بشكل رائع عندما يكون هنالك أكثر من طلب في نفس الوقت.

العبارة SELECT

يمكنك الحصول على البيانات في صورة سجلات باستخدام لغة SQL وذلك عن طريق العبارة SELECT والتي تأخذ الشكل العام التالي :

```
SELECT fields FROM tables;
```

حيث أن fields هي أسماء الحقول وtables هي أسماء الجداول التي نريد أن نحصل على الحقول منها، فإذا كان لدينا الجدول التالي في قاعدة البيانات على سبيل المثال :

Directory Table	
Telephone	Name
1291233	John
1682340	Tim

Jim	2462466
Dill	3636778

إذا أردنا أن نحصل على الحقلين Name و Telephone في الجدول السابق فإن العبارة التي سنستخدمها هي : `SELECT name,telephone FROM directory;`

والعبارة السابقة تعيد الجدول كما هو في الأعلى بالضبط، ولكن ماذا لو كتبنا كلمة telephone قبل كلمة name في عبارة SQL السابقة فتصبح كالتالي : `SELECT telephone,name FROM directory;`

عند تنفيذ العبارة السابقة سنحصل على الجدول التالي :

telephone,name	
Telephone	Name
1291233	John
1682340	Tim
2462466	Jim
3636778	Dill

أما إذا كتبنا العبارة كالتالي : `SELECT name,telephone,name FROM directory;`

فسنحصل على النتيجة التالية :

name,telephone,name		
Name	Telephone	Name
John	1291233	John
Tim	1682340	Tim
Jim	2462466	Jim
Dill	3636778	Dill

وماذا لو قمنا بتنفيذ العبارة التالية : `SELECT name FROM directory;`

فالناتج سيكون كالتالي :

name
Name
John
Tim
Jim
Dill

اما في حالة وجود تكرار في الحقول فاننا نستطيع حذف التكرار في العرض كالتالي :

```
Select Distinct name,telephone  From directory;
```

التريتب التنازلي و التصاعدي :

أولا للترتيب التصاعدي نستخدم الأمر ORDER BY كالتالي:

```
Select Distinct name,telephone From directory
    ORDER BY telephone ;
```

أما للترتيب التنازلي فإننا نضيف الجزء DESC للمثال السابق:

```
Select Distinct name,telephone From directory
    ORDER BY telephone DESC;
```

كما تلاحظ ليست هنالك أية قواعد ثابته، فلا يمكنك أن تقول بأن الحقل name هو الحقل الأول في الجدول، أنت من يحدد الآن ما هو الحقل الأول وما هو الحقل الثاني وهكذا.

قد تحتاج في بعض الأحيان أن تعرض جميع الحقول في الجدول، فيمكنك استخدام علامة النجمة (*) في مكان الحقول للحصول على جميع الحقول التي في الجدول، فاستخدام العبارة التالية سيعيد الجدول بأكمله كما كتبناه أول مرة : `SELECT * FROM directory;`

قواعد اللغة

عند كتابتك لاستعلامات SQL يجب أن تتذكر الأمور التالية دائما :

- لغة SQL لا تفرق بين الحروف الكبيرة والصغيرة فلا فرق بين كتابة الكلمة SELECT والكلمة select والكلمة SeLeCt وكلها تعامل بنفس الطريقة، قد تكون هنالك بعض الاستثناءات في أسماء الجداول أو الحقول، يجب أن تراجع دليل الاستخدام المرفق مع برنامج قاعدة البيانات الذي تستخدمه للتأكد من ذلك.
- المسافات البيضاء ليس لها اعتبار في لغة SQL، فيمكنك وضع أي قدر تريد من المسافات البيضاء لتنسيق استعلاماتك، فيمكنك مثلا أن تكتب الاستعلام في الصورة التالية، ولا توجد أية مشكلة في ذلك :
- SELECT   *
- FROM directory;
- تنتهي جميع الاستعلامات بالفاصلة المنقوطة (؛).
- العبارات النصية التي لا تعتبر جزءا من عبارات الاستعلامات توضع بين قوسي اقتباس مفردين ويتيح بعض برامج قواعد البيانات استخدام أقواس الاقتباس المزودجة أيضا (" .. ").

- إذا كنت تريد استخدام علامات الاقتباس كجزء من النص الذي تريد إدخاله إلى قاعدة البيانات فإنك تضع قبلها علامة الشرطة الخلفية لتصبح هكذا ('\ ) أو في بعض برامج قواعد البيانات فيتم ذلك بمضاعفة علامة الاقتباس (")، فمثلا إذا أردت وضع العبارة التالية في قاعدة البيانات : `I'm me`

فإنك تكتبها في أحد الصور التالية حسب برنامج قاعدة البيانات الذي تستخدمه :

'I\'m me'
'I''m me'

الشروط

يمكنك أثناء جاب السجلات أن تضع شروطا معينة للسجلات التي تريد الحصول عليها بواسطة العبارة SELECT وذلك باستخدام المقطع WHERE وبعده تضع الشروط التي تريدها، أنظر مثلا : `SELECT telephone FROM directory WHERE name='Tim';`

فإن ناتج تنفيذ الاستعلام السابق سيكون كالتالي :

name='Tim'
telephone
1682340

حيث أن هناك حقلا واحد فقط يطابق الشرط، والشرط هو أن يكون الاسم name يساوي Tim ولأن كلمة Tim جزء من البيانات المخزنة في قاعدة البيانات فإننا نحيطها بعلامات الاقتباس المفردة، لاحظ أيضا أن البرنامج لن يعيد الاسم أيضا ضمن النتائج وذلك لأننا لم نطلب الاسم في عبارتنا، ويمكننا الحصول على الاسم أيضا باستخدام هذه العبارة : `SELECT telephone, name FROM directory WHERE name='Tim';`

كما تلاحظ فإن عبارة SELECT من بدايتها وحتى ما قبل كلمة WHERE تعمل كما شرحنا في السابق بالضبط.

الجزء الذي يهمنا الآن في العبارة هو الجزء الذي يأتي بعد الكلمة WHERE أو ما يسمى بالشرط condition، تتكون عبارة الشرط الواحدة من ثلاثة أجزاء، الجزء الأول هو الطرف الأيسر من العبارة والجزء الثاني هو الطرف الأيمن من العبارة والجزء الثالث هو

المعامل الذي يقع بين الطرفين، والمعامل في مثالنا السابق هو علامة المساواة (=) حيث أن شرطنا هو أن يكون الطرف الأيمن *يساوي* الطرف الأيسر حتى يتحقق الشرط :

rightside	operator	leftside
'Tim'	=	name

والعوامل المستخدمة في الشروط مختلفة، أهما ما يلي :

معاملات الاختبار فيSQL	
المعامل	**اسمه**
=	يساوي
>	أكبر من
<	أصغر من
<=	أكبر من أو يساوي
>=	أصغر من أو يساوي
<>	لا يساوي
LIKE	يشبه

العامل الأول هو عامل المساواة وهو يتحقق عندما يكون الطرف الأيمن والأيسر متساويان، كما رأينا في المثال السابق، العوامل التالية تبدو واضحة وهي > ويتحقق عندما يكون الطرف الأيسر أكبر من الطرف الأيمن، < ويتحقق عندما يكون الطرف الأيمن أصغر من الطرف الأيسر، وبعدها أكبر من أو يساوي ثم أصغر من أو يساوي.

أما المعامل السادس <> فيعني لا يساوي ويتحقق الشرط فيه عندما يكون الطرف الأيمن لا يساوي الطرف الأيسر.

قد تبدو مقارنة النصوص باستخدام العوامل < و > غريبة نوعا ما، فكيف نقول مثلا بأن 'Tim' < 'Jim' ؟ الأمر في غاية البساطة، كل ما عليك فعله هو أن تتخيل بأنك تريد أن ترتب هذا الجدول تنازليا فهذا يعني بأن القيم العليا تكون فوق والقيم الدنيا تكون في أسفل الترتيب، ولهذا فإن 'z' < 'a' تعتبر عبارة صحيحة.

العامل الأخير من عوامل المقارنه هو العامل LIKE (يشبه) وهو يستخدم لمقارنة النصوص عادة، ويتحقق الشرط فيه عندما يكون الطرف الأيمن يشبه الطرف الأيسر، ويكتب طرفها الأيمن في صورة نص يحتوي على علامات النسبة المؤوية (%) وهي تعني (أي شيء) بمعنى أنك إذا قلت : `SELECT name,telephone FROM directory WHERE name LIKE '%m';`

فهذا يعني بأن الاسم يجب أن يكون (أي شيء) ثم الحرف 'm'، أو بمعنى آخر سيكون الشرط متحققا في السجلات التي ينتهي الاسم فيها بالحرف 'm'، وإذا كتبنا : `SELECT name,telephone FROM directory WHERE name LIKE 'm%';`

فهذا يعني m ثم (أي شيء) أي أنها تطابق حقول name التي تبدأ بالحرف m، حسنا ماذا لو قلنا : `SELECT name,telephone FROM directory WHERE name LIKE '%m%';`

أما هذه فتعني (أي شيء) ثم الحرف m ثم (أي شيء) أي أنها ستطابق جميع السجلات التي يحتوي الحقل name فيها على الحرف m.

يمكنك أن تقيس على ذلك الكثير من الأمور، فيمكنك أن تبحث في قاعدة البيانات عن حقل يبدأ بكلمة 'this' ويبعدها بعدة أحرف أو كلمات أو (أي شيء) تأتي كلمة 'one' فتكتب هكذا : `text LIKE 'this%one%'`

دعنا نجرب تطبيق هذا الاستعلام على قاعدة البيانات التي نعمل عليها :

```
SELECT name,telephone FROM directory
WHERE name LIKE '%m';
```

فإن النتائج ستكون كالتالي :

name LIKE '%m'	
Name	**Telephone**
Tim	1682340
Jim	2462466

استخدام أكثر من شرط

يمكنك استخدام أكثر من شرط واحد عن جلب سجلات بياناتك، فيمكنك مثلا أن تبحث عن جميع الحقول التي يبدأ اسم صاحبها بالحرف J والحرف D، أو ربما تريد البحث عن جميع الأشخاص الذين تاريخ ميلادهم أكبر من ١ يناير ٢٠٠٠ وأصغر من ٥ فبراير ٢٠٠٠ وهكذا أمور، يمكنك أن تربط بين الشروط باستخدام أداتين مختلفتين للربط هما AND و OR، وهاتان الأداتان تساويان && و || على التوالي في بعض لغات البرمجة.

الأداة الأولى ومن اسمها AND (و) تجعل السجل محققا للشرط عندما يتحقق الشرط الذي على يمينها والشرط الذي على يسارها معا، فمثلا عندما نقول :

```
SELECT name,telephone FROM directory WHERE
name LIKE '%m' AND telephone > 20000000;
```

ويعني ذلك بأن السجلات الوحيدة التي سيتم عرضها هي التي يتحقق فيها كل من الشرطان معا، فتكون name تنتهي بالحرف m ورقم الهاتف أكبر من ٢٠٠٠٠٠٠٠، فهي تعيد السجل الوحيد الذي يحقق الشرطان كالتالي :

name LIKE '%m' AND telephone > 20000000	
Name	**Telephone**
Jim	2462466

أما OR (أو) فيكفي لتحققها أن يتحقق أحد الشرطان فقط، فإذا قلنا :

```
SELECT name,telephone FROM directory WHERE
name LIKE '%m' OR telephone > 20000000;
```

فإن جميع الحقول التي ينتهي فيها الحقل name بالحرف m بالإضافة إلى جميع الحقول التي يكون فيها الحقل telephone أكبر من ٢٠٠٠٠٠٠٠، أي أنها تعيد جميع الحقول التي تحقق الشرط الأول، وجميع الحقول التي تحقق الشرط الثاني وجميع الحقول التي تحقق الشرطان معا، ولذلك فإن ناتج تنفيذ العبارة هو ثلاثة سجلات كالتالي :

name LIKE '%m' OR telephone > 20000000	
Name	**Telephone**

Jim	2462466
Tim	1682340
Dill	3636778

كما تلاحظ، السجل الأول Tim كان من ضمن السجلات التي حققت الشرطان معا فكان من ضمن جدول النتائج، أما السجل الثاني Jim فلم يحقق الشرط الثاني وكان الحقل telephone فيه أقل من ٢٠٠٠٠٠٠٠ إلا أنه كان من ضمن جدول النتائج لأنه حقق الشرط الأول وهذا كاف، والسجل الثالث Dill حقق الشرط الثاني ولم يحقق الأول حيث أنه لا ينتهي بالحرف m.

استخدام OR و AND معا

يمكنك ربط أكثر من شرط باستخدام أكثر من أداة ربط ولأكثر من مرة، فيمكنك أن تقول مثلا :

```
SELECT name,telephone FROM directory WHERE
telephone > 20000000 OR telephone = 30000000 AND name LIKE '%m';
```

العبارة السابقة تنطوي على حيلة ما، قد يبدو لك في الوهلة الأولى أن العبارة تقوم بمطابقة جميع السجلات التي يكون الحقل name فيها ينتهي بالحرف m وفي نفس الوقت يكون رقم الهاتف فيها أكبر من ٢٠٠٠٠٠٠٠ أو يكون يساوي ٣٠٠٠٠٠٠٠، ولكن الواقع يختلف عن ذلك، عندما يكون هنالك أكثر من أداة ربط، ويكون هنالك أكثر من شرطين، فإن برنامج قاعدة البيانات سيقوم بتنفيذ أداة الربط AND أولا حسب ترتيبها بالجملة، وبعد ذلك يقوم بربط الجمل التي تستخدم الأداة OR.

يعني ذلك بأنه في العبارة السابقة الأداة AND تربط شرطان هما telephone = ٣٠٠٠٠٠٠٠ و 'name LIKE '%m أما الأداة OR فتربط شرطان هما telephone > ٢٠٠٠٠٠٠٠ و 'telephone = 30000000 AND name LIKE '%m، ونقول هنا بأن AND لها أولوية التنفيذ حيث قامت بربط الشروط التي على جانبيها، وأصبح شرطا واحدا وبعد ذلك يبدأ عمل الأداة OR في ربط الشروط التي على جانبيها واللذان كانا الشرط الكبير الناتج من ربط الأداة AND للشرطان الصغيران، بالإضافة إلى الشرط الآخر الذي على شمالها.

ماذا لو أردنا الآن أن نجبر برنامج قاعدة البيانات على تنفيذ الشرط OR أولا ؟ لعمل ذلك نقوم بإحاطة الشرطان اللذان على جانبا الأداة OR بالأقواس، فتصبح العبارة هكذا :

```
SELECT name,telephone FROM directory WHERE
(telephone > 20000000 OR telephone = 30000000) AND name LIKE '%m';
```

لأن الأقواس لها أولوية أعلى في التنفيذ من العبارة OR (الأقواس لها أعلى أولوية على الإطلاق)، فإن البرنامج يقوم بتنفيذ ما بداخل الأقواس أولا، وفي داخل الأقواس سيجد شرطان مربوطان بالأداة OR فيقوم بربطهما وينتج شرط واحد كبير، وبعد ذلك يأتي دور العبارة AND فترتبط الشرط الكبير الذي بداخل الأقواس مع الشرط الذي يأتي على يمينها وهو 'name LIKE '%m.

لاحظ أن الجزء الذي أتى بعد العبارة OR داخل الأقواس غير ضروري أبدا لأن الشرط سيتحقق دائما إذا كان الرقم يساوي ٣٠٠٠٠٠٠٠ حتى لو لم نطلب منه ذلك تحديدا، حيث أن الشرط الذي على يسار العبارة يكفي لذلك (تذكر بأن ٣٠٠٠٠٠٠٠ أكبر دائما من ٢٠٠٠٠٠٠٠)، ولكننا أوردناها هنا كمثال فقط.

النفي بالعبارة NOT

تقوم العبارة NOT بنفي الشرط الذي يأتي بعدها، فتصبح العبارة صحيحة إذا كانت خاطئة وتصبح خاطئة إذا كانت صحيحة، مثلا العبارة التالية :

```
SELECT name,telephone FROM directory WHERE NOT name = 'Tim';
```

ستعيد جميع السجلات التي لا يكون فيها الحقل name يساوي Tim وهكذا، ويمكن استخدامها في تراكيب أكثر تعقيدا مع الأداتان AND و OR والأقواس.

الان نعتبر أن لدينا جدول بيانات و لنعتبر أن الجدول اسمه ☹ Course) و هو يحتوي مثلا على معلومات عن طلبة مسجلين لمادة ما و الجدول يحتوي على معلومات لهم .

الجدول كالتالي :

Course Table :

Std_name	Std_id	City	Date_of_birth	Telephone
Ali	2014	Abu Dhabi	13/1/1980	2073456
Fares	1234	Dubai	21/8/1982	2845632
Ahmed	547	Al Ain	13/6/1985	2014789

<u>استخدام الدوال في الاستعلامات</u>

سنبدأ بعمل استعلامات تستخدم دوالا مثل المجموع و المعدل و العداد و القيم الأصغر و الأكبر و الدوال كالتالي :

١١ Sum() Function
١٢ Count() Function
١٣ Max() Function
١٤ Min() Function
١٥ Avg() Functi

لنفرض أننا نود عرض عدد الطلبة المسجلين للمادة . للعمل نستخدم الاستعلام التالي :

```
Select count(*)   As   [Student_number]

From Course ;
```

الناتج كالتالي :

Student_number
3

و بالتالي يقوم الاستعلام بعمل جدول جديد به حقل واحد اسندنا اليه اسم ((Student_number يعرض عدد الطلبة الذي قد عدهم .

مثال تطبيقي اخر على الدوال :

```
Select Min(age) AS [Minimum Age] ,Max(age) AS [Maximum Age],Avg(age) AS average

From Course ;
```

الناتج كالتالي :

Minimum Age	Age Maximum	average
17	22	19.67

ادخال الحقول

في هذا الدرس سنبدأ بعمل استعلامات تمكننا من إضافة سجل أي صف للجدول الموجود بقاعدة البيانات ...

الجدول كالتالي :

Course Table :

Std_name	Std_id	City	Date_of_birth	Telephone	Age
Ali	2014	Abu Dhabi	13/1/1980	2073456	22
Fares	1234	Dubai	21/8/1982	2845632	20
Ahmed	547	Al Ain	13/6/1985	2014789	17

لنفرض أننا نود ادخال حقل أي أننا سنقوم بادخال معلومات لطالب جديد من خلال جمل الاستعلام .. ماذا نفعل ؟؟ دعونا نرى :

```
INSERT INTO Course

VALUES ( 'Khalid'  , '1221' , 'Sharjah' , '12/4/1985' , '2020202' , '17'  ) ;
```

الناتج كالتالي :

Std_name	Std_id	City	Date_of_birth	Telephone	Age
Ali	2014	Abu Dhabi	13/1/1980	2073456	22
Fares	1234	Dubai	21/8/1982	2845632	20
Ahmed	547	Al Ain	13/6/1985	2014789	17
Khalid	1221	Sharjah	12/4/1985	2020202	17

لنفرض أننا نود ادخال حقل أي أننا سنقوم بادخال معلومات لطالب جديد من خلال جمل الاستعلام .. ماذا نفعل ؟؟ دعونا نرى :

```
INSERT INTO Course (Std_name , Age )

VALUES ( 'Khalid'  , '17'  ) ;
```

الناتج كالتالي :

Std_name	Std_id	City	Date_of_birth	Telephone	Age
Ali	2014	Abu Dhabi	13/1/1980	2073456	22
Fares	1234	Dubai	21/8/1982	2845632	20
Ahmed	547	Al Ain	13/6/1985	2014789	17
Khalid					17

و من الملاحظ أنه تم ادخال الاسم و العمر فقط و التي حددناها .

تعديل الحقول

في هذا الدرس سنبدأ بعمل استعلامات تستخدم في تعديل بيانات حقل موجود بغرض تصحيح الخطأ أو الحاجة نتيجة التغيرات على البيانات ..... و لنفرض أن الجدول الموجود هو التالي :

Course Table :

Std_name	Std_id	City	Date_of_birth	Telephone	Age
Ali	2014	Abu Dhabi	13/1/1980	2073456	22
Fares	1234	Dubai	21/8/1982	2845632	20
Ahmed	547	Al Ain	13/6/1985	2014789	17

لنفرض أننا نود تعديل بيانات علي في الجدول السابق و لنفرض ان الهاتف قد تغير رقمه و نود أن نغيره .. لعمل ذلك نفعل التالي :

```
UPDATE  Course  SET    Telephone= '2020202'
WHERE  std_name = 'Ali'
```

الناتج كالتالي :

Std_name	Std_id	City	Date_of_birth	Telephone	Age
Ali	2014	Abu Dhabi	13/1/1980	2020202	22

Fares	1234	Dubai	21/8/1982	2845632	20
Ahmed	547	Al Ain	13/6/1985	2014789	17

و لتغيير أكثر من عمود نفعل التالي :

```
UPDATE  Course
SET city = 'Sharjah', Age = '23'
WHERE  Std_name = 'Ali'
```

الناتج كالتالي :

Std_name	Std_id	City	Date_of_birth	Telephone	Age
Ali	2014	Sharjah	13/1/1980	2020202	23
Fares	1234	Dubai	21/8/1982	2845632	20
Ahmed	547	Al Ain	13/6/1985	2014789	17

الحذف من الحقول

سنبدأ بعمل استعلامات تستخدم في حذف حقل من الحقول الموجودة في قاعدة البيانات حسب الشرط الموجود مع الاستعلام و الذي يقوم بتحديد السجل المراد حذفه .. و لنفرض أن الجدول الموجود هو :

Course Table :

Std_name	Std_id	City	Date_of_birth	Telephone	Age
Ali	2014	Abu Dhabi	13/1/1980	2073456	22
Fares	1234	Dubai	21/8/1982	2845632	20
Ahmed	547	Al Ain	13/6/1985	2014789	17

درس الحذف بسيط جدا و تتلخص عمليته في الاستعلام التالي :

```
DELETE   FROM  Course  WHERE city = 'Abu Dhabi'
```

الناتج كالتالي :

Std_name	Std_id	City	Date_of_birth	Telephone	Age
Fares	1234	Dubai	21/8/1982	2845632	20
Ahmed	547	Al Ain	13/6/1985	2014789	17

انشاء الجداول قواعد البيانات و

في هذا الدرس سنتعلم كيف ننشيء قاعدة بيانات و كيف نضيف اليها الجداول وفي حقيقة الأمر فإنها ليست ضرورية نوعا ما لكن من باب العلم و الإحاطة ......

و لنبدأ باستعلام يقوم بتعريف أو انشاء قاعدة للبيانات كالتالي :

```
CREATE DATABASE  School
```

و هنا قمنا بتعريف قاعدة بيانات تحمل اسم School و لكي ننشيء فيها الجداول نكتب الاستعلام التالي :

```
CREATE TABLE Students
  (
    Name varchar,
    City varchar,
    Address varchar,
    Age int
   )
```

وهنا قد قمنا بكتابة استعلام وظيفته انشاء جدول يسمى School يحوي معلومات عن الطلبة و قد حددنا نوعية البيانات لكل خانة سواء كانت أعدادا أو سلسلة حرفية أي نصا .....

و بما أنك قد وصلت هنا فإنك قد ملكت نظرة واسعة عن لغة الاستعلام التي ستفيدك جيدا في تكنولوجيا Asp.net و التي تعتمد على قواعد البيانات و تطبيقاتها ..

النماذج FORMS

النموذج عبارة عن مستند يشتمل على بيانات سجل واحد بحيث تظهر بشكل جذاب مع إمكانية التحكم في كل عنصر من عناصره .

ويتم إنشاء النماذج بثلاث طرق هي :-

- انشاء نموذج تلقائي : ويتم انشاءه بالطرق التالية :-

أ) من إطار قاعدة البيانات ، نشط التبويب "نماذج" ثم انقر زر "جديد" يظهر مربع نموذج جديد .

ب) يحتوي هذا المربع على ٣ أشكال من النماذج التلقائية وهي :

١. نموذج تلقائي عامودي : حيث تظهر الحقول في عمود واحد .

٢. نموذج تلقائي جدولي : حيث تظهر الحقول على شكل جدول .

٣. نموذج تلقائي صفحة بيانات : حيث تظهر على شكل صفحة البيانات .

جـ) اختر النموذج التلقائي الذي تريد ثم حدد الجدول الذي سوف تأخذ منه الحقول .

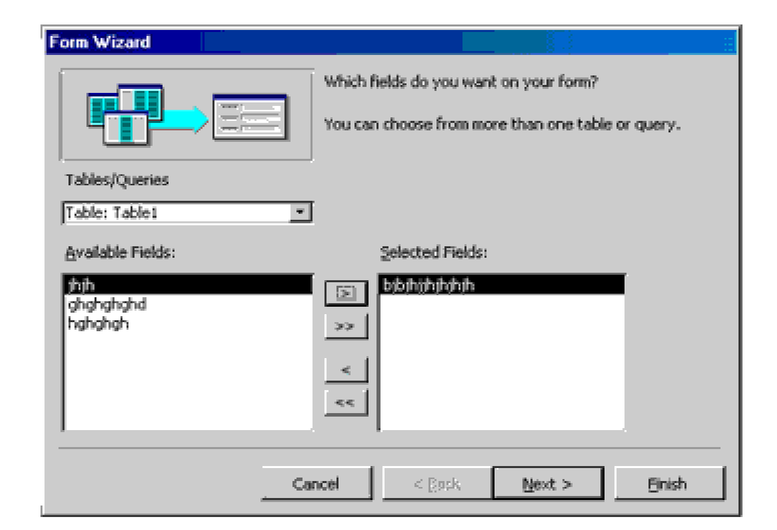
د) انقر الزر موافق .

انشاء نموذج باستخدام معالج النماذج :

١. من إطار قاعدة البيانات ، نشط التبويب "نماذج" ثم انقر زر جديد فيظهر مربع "نموذج جديد" .

٢. انقر معالج النماذج وحدد الجدول الذي سوف تأخذ منه الحقول .

٣. انقر الزر موافق يظهر مربع معالج النماذج .



٤. اختر الحقول التي تريدها من قائمة الحقول المتاحة وقم بإضافتها إلى قائمة الحقول المحددة وذلك عن طريق الزر ***. بعد ذلك انقر زر التالي .

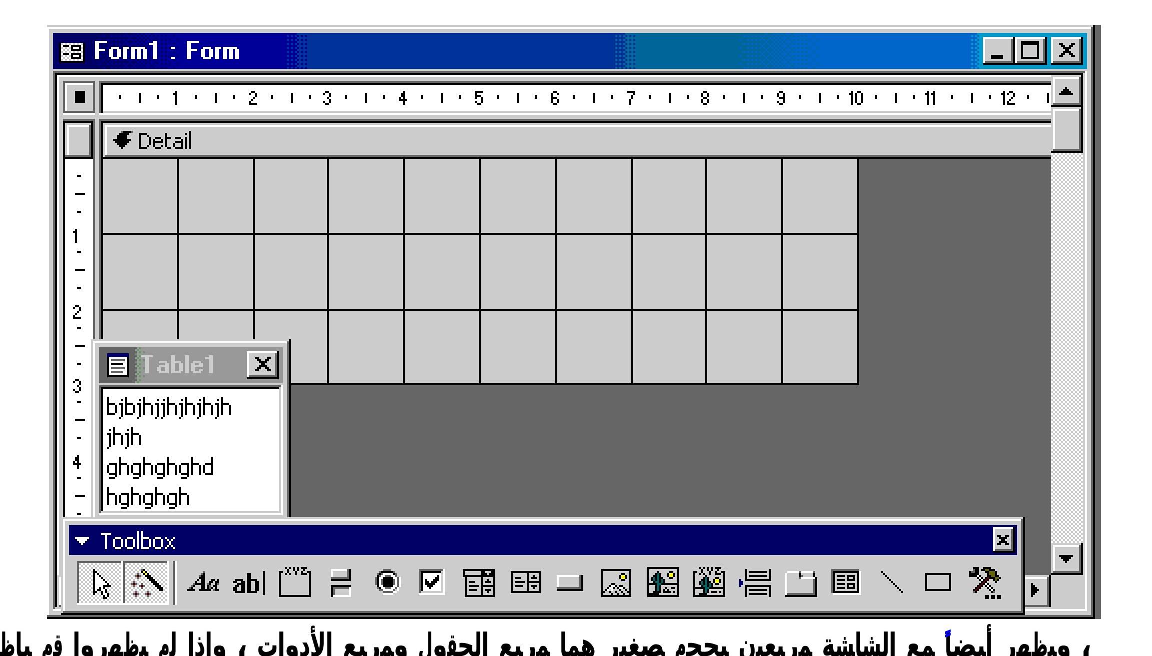
٥. يظهر مربع آخر يطلب منك تحديد نوع التخطيط الذي تريده ، اختر النوع الذي تريده ثم انقر زر التالي . يظهر مربع حواري آخر يطلب منك كتابة عنوان للنموذج ، اكتب العنوان ثم انقر زر انهاء .

إنشاء نموذج بطريقة عرض التصميم (يدوي) :

١. من إطار قاعدة البيانات نشط التبويب "نماذج" ثم انقر زر جديد فيظهر مربع نموذج جديد . (انظر الشكل)

٢. انقر طريقة عرض التصميم ، وحدد الجدول الذي سوف تأخذ منه الحقول .

٣. انقر زر موافق ، فتظهر شاشة تصميم النموذج



، ويظهر أيضاً مع الشاشة مربعين بحجم صغير هما مربع الحقول ومربع الأدوات ، وإذا لم يظهروا قم بإظهارها عن طريق الضغط على زر قائمة الحقول وزر مربع الأدوات على شريط الأدوات .

٤. قم بسحب الحقول التي تريدها من قائمة الحقول إلى شاشة تصميم النموذج في قسم تفصيل .

٥. قم بحفظ النموذج .

تستطيع أن تتحكم في معظم العناصر الموجودة داخل شاشة تصميم النموذج مثل إضافة حقول جديدة أو نقل حقل إلى مكان آخر أو ترتيب الحقول أو إضافة نص إلى الحقل أو إضافة رأس وتذييل إلى النموذج أو تغيير حجم الأقسام وحجم الحقول وإضافة عناصر تحكم داخل النموذج (أزرار أوامر) خانات اختيار ، مربعات كتابة ، مربعات سرد ، مربعات كتابة سرد .

إضافة عناصر التحكم :

يتعامل برنامج Access مع عناصر التحكم بصفة مستقلة أي أن كل عنصر يعامل ككائن مستقل ولذلك يمكن اختياره وسحبه ونقله متى شئت .

ويتم إضافة عناصر التحكم من شريط مربع الأدوات الذي يظهر في نافذة تصميم النموذج .

١. انشاء خانة اختيار(Check Box (: تستخدم هذه الخانه لاختيار قيمة من اثنين (نعم) أو (لا) ولانشائها :

أ) انقر زر خانة الاختيار الموجود في مربع الأدوات حيث يتحول المؤشر إلى مربع مصحوب بعلامة زائد حدد المكان الذي سوف تضع به خانة الاختيار فيتم ربط هذه الخانة بالحقل المنطقي لها .

كما يمكن سحب حقل منطقي من نافذة الحقول ووضعه داخل النموذج فيقوم البرنامج بربطه بالخانة بحيث يظهر عنوان الحقل على يمين خانة الاختيار .

ب) إضافة مربع تحرير وسرد Box Combo : يشتمل هذا المربع على جزأين بجزء تكتب فيه القيمة التي تريدها وجزء يحتوي على قائمة تختار منها ما تريد . ويتم انشاءه كما يلي :

١. قم بإظهار زر مربع الأدوات وزر قائمة الحقول من شريط الأدوات إذا لم يكونوا ظاهرين .

٢. تأكد أن زر معالجات عناصر التحكم مختاراً في مربع الأدوات .

٣. انقر زر مربع التحرير والسرد الموجود في مربع الأدوات حيث يتحول المؤشر إلى مربع مصحوباً بعلامة + .

16 ضع المؤشر أمام الحقل الذي تريده في شاشة تصميم النموذج ، أو قم بسحب حقل من قائمة الحقول إلى نافذة شاشة التصميم فيظهر مربع حوار معالج التحرير والسرد .

17 حدد الخيار الذي تريده ثم تابع مربعات الحوار التي تظهر حتى النهاية .

بعد ضغط زر إنهاء يغلق مربع الحوار وترجع إلى نافذة تصميم النموذج قم بسحب مربع التحرير والسرد إلى المكان المناسب .

جـ) إنشاء أزرار تبديل أو مجموعة خيار

١. اختر طريقة تصميم النموذج ، ثم انقر على زر مجموعة الخيار يتحول المؤشر إلى مربع مصحوب بعلامة + .

٢. ضع المؤشر أمام الحقل الذي تريده في شاشة التصميم فيظهر مربع معالج مجموعة الخيار .

٣. قم بكتابة الخيارات التي تريدها ثم انقر زر التالي ثم تابع مربعات الحوار حتى النهاية .

انقر على زر إنهاء يغلق مربع الحوار وترجع إلى نافذة التصميم (انقل العنصر إذا لم يعجبك مكانه) بدل إلى طريقة عرض النموذج لكي تشاهد النتيجة .

العمليات الحسابية في النماذج

لإجراء أي عملية حسابية في النماذج كالتالي

١. يتم فتح النموذج على التصميم
٢. يتم الضغط على الحقل المراد إيجاد الناتج به بيمين الماوس ثم نختار خصائص ثم من مصدر عنصر التحكم في باب بيانات نضغط على الزر منشأ التعبير
٣. يتم حذف الكلمة القديمة ثم يتم الضغط مرتين متتاليتين على دالات ثم على وظائف مضمنة يتم الضغط مرة واحدة ثم نختار العملية الحسابية المطلوبة مثل sum او average المعدل (avg) نضغط مرتين متتاليتين .
٤. ثم نضغط على expr مرة
٥. ثم نختار الحقل المراد إيجاد الناتج له مرتين متتاليتين ثم موافق ثم يتم التنفيذ .

ولعمل العمليات الحسابية بطريقة المعادلة

٦. يتم فتح النموذج على التصميم
٧. ثم في الحقل المراد إيجاد الناتج به يتم الضغط عليه بيمين الماوس ثم خصائص ثم من مصدر عنصر التحكم .....نختار منشأ التعبير
٨. يتم كتابة =
٩. ثم التأشير على الحقل المطلوب مرتين متتاليتين ثم كتابة العلاقة الرياضيه المطلوبة مثل + / * - الخ ....والتأشير على الحقل الأخر مرتين أيضا لادراجة في المعادلة وتكرار العملية حسب الحقول ثم يتم التنفيذ.

مثال

<u>لإيجاد ناتج الراتب –الخصم</u>

١. يتم فتح النموذج على التصميم .
٢. يتم الضغط بيمين الماوس على حقل الناتج ثم نختار خصائص ثم الضغط على منشأ التعبير
٣. يتم كتابة = ثم الضغط على حقل الراتب مرتين وكتابة – والضغط مرتين على حقل الخصم ثم يتم التنفيذ

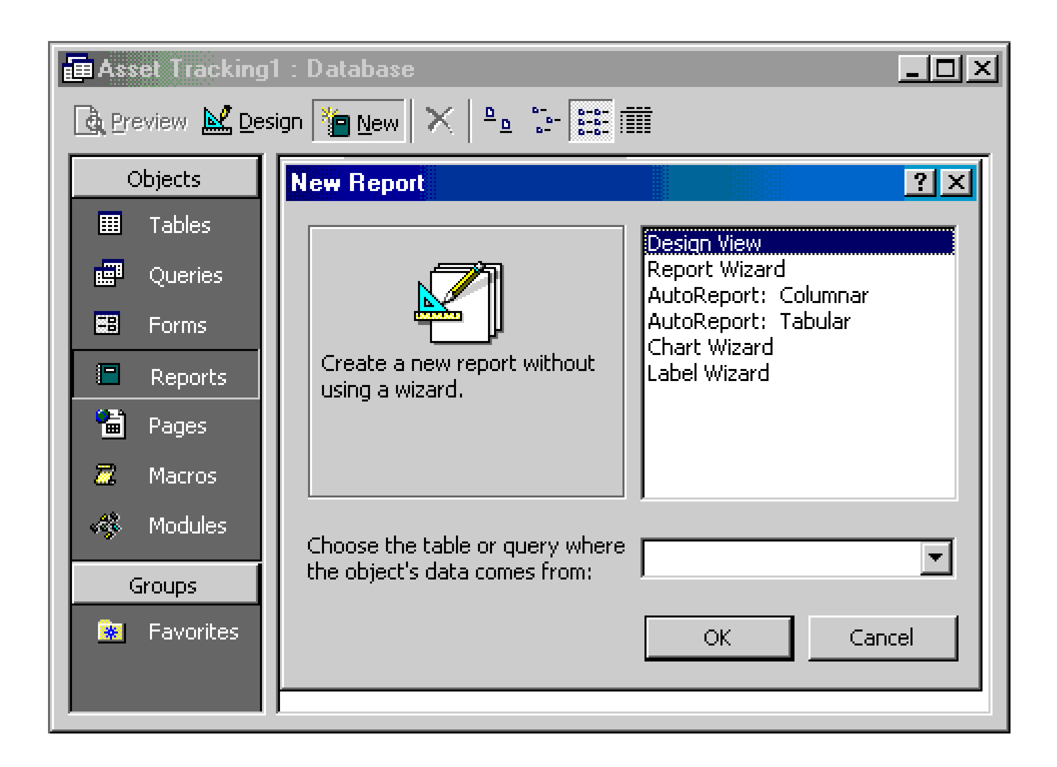
التقاريرReports وبطاقات التسمية

التقارير عبارة عن مستند يمكن طباعته أو عرضه على شاشة الكمبيوتر أو حفظه في ملف :

توجد ٣ طرق لإنشاء التقارير وهي :-

إنشاء تقرير تلقائي : 18

١. من إطار قاعدة البيانات ، نشط التبويب (تقارير) ثم انقر زر (جديد) فيظهر مربع (تقرير جديد) .



٢. اختار تقرير تلقائي عمودي أو جدولي . ثم حدد الجدول الذي سوف نأخذ منه الحقول .

٣. انقر الزر موافق .

انشاء تقرير باستخدام معالج التقارير :-

١. من إطار قاعدة البيانات اختر التبويب تقرير ثم انقر زر (جديد) فيظهر مربع (تقرير جديد) .

٢. اختار معالج التقارير وحدد الجدول الذي سوف تأخذ منه الحقول .

٣. انقر الزر موافق ، فيظهر مربع معالج التقارير .

٤. اختار الجدول الذي تريده من خانة جداور / استعلامات .

٥. انقل الحقول التي تريدها من خانة الحقول المتاحة إلى خانة الحقول المحددة بواسطة الزر *** ثم انقر الزر التالي .

فيظهر مربع حوار آخر يسألك عن رغبتك في إضافة مستويات تجميع للتقرير إذا رغبت بذلك قم بتحديد الحقول وإضافتها بواسطة الزر *** ، ومن ثم نقر زر خيارات تجميع . أما إذا لم ترغب فاضغط على زر التالي ، فيظهر مربع حوار آخر فيسألك ما هو الترتيب الذي تريده للسجلات ، ضع الترتيب الذي تريده ثم انقر زر التالي . فيظهر مربع حوار آخر يطلب منك تحديد نمط التقرير واتجاه الطباعة .

حدد الذي تريده ثم انقر زر التالي ، فيظهر مربع حوار آخر يطلب منك إدخال عنوان للتقرير ، اسحب 19
العنوان ثم انقر زر إنهاء .

<u>إنشاء تقرير بطريقة عرض التصميم :</u>

١. من إطار قاعدة البيانات نشط التبويب (تقارير) ثم انقر زر جديد فيظهر مربع (تقرير جديد) .

٢. اختار طريقة عرض التصميم ثم حدد الجدول الذي سوف تأخذ منه المعلومات .

٣. انقر زر موافق .

٤. تظهر شاشة تصميم النموذج حيث تحتوي على قسم تفصيل وتحتوي أيضاً على مربع قائمة حقول .

اسحب الحقول التي تريدها إلى قسم تفصيل وقم بعد ذلك بعمل ما تريده كما تم طرحه في إنشاء نموذج بطريقة عرض التصميم .
كما تم طرحه في إنشاء نموذج بطريقة عرض التصميم .

<u>إنشاء بطاقات التسمية (الملصقات) :</u>

وهي عبارة عن بطاقات تستخدم غالباً في طباعة عنوان وذلك لاستخدامه كلاصق على مظروف معين . ويتم انشائه كما يلي :

١. من نافذة قاعدة البيانات نشط التبويب "تقارير" ثم انقر زر جديد فيظهر مربع تقرير "جديد" .

٢. اختار معالج التسمية ثم حدد الجدول الذي سوف تأخذ منه المعلومات .

٣. انقر زر موافق ، فيظهر مربع معالج التسمية .

٤. اختر حجم التسمية ونوعها ثم انقر زر التالي فتظهر نافذة أخرى حدد منها نوع الخط وحجم الخط ... الخ ثم انقر زر التالي .

٥. يظهر مربع حواري آخر يطلب منك إضافة الحقول الموجودة في خانة الحقول المتاحة ووضعها في خانة النموذج الأولي للتسمية وذلك عن طريق الزر > .

٦. انقر زر التالي ، فيظهر مربع حواري آخر يطلب منك الحقول التي ترغب في إجراء ***** . حدد الحقول ثم انقر زر التالي فيظهر مربع حواري آخر يطلب منك عن كتابة اسم للتقرير .

٧. اكتب الاسم ثم انقر زر انهاء .

٨. اغلق نافذة المعاينة ، فيتم حفظ تقرير التسمية .

قواعد البيانات Microsoft Access

الماكرو

-

تعريف الماكرو : هو سلسلة من العمليات التي تنفذ كامر واحد الهدف منه السرعة والسهوله للوصول الى غرض ما .

هنا سوف استعرض بالشرح المبسط جدا لعملية إنشاء الماكرو

يتم عمل الماكرو بالضغط على زر وحدات الماكرو في قاعدة البيانات ثم جديد ونختار الاجراء المناسب من القائمة المنسدلة للاجراءات يمكنك اختيار الاجراء(وهذه بعض الاجراءات) :-

Beep	إصدار صوت
Close	إغلاق
Go to record	للتنفل بين السجلات التالي /السابق /الاول الخ......
msgbox	لوضع رسالة
Maximize	تكبير
Minimize	تصغير
quit	للخروج من البرنامج
open form	لفتح نموذج
open query	لفتح استعلام
open table	لفتح جدول
Print out	طباعة
runapp	تشغيل برنامج مثل pbrushعلى سبيل المثال

مثلا :- عند اختيار الإجراء go to record يجب اختيار الكائن المراد ربط الماكرو به مثل النموذج او استعلام او جدول ....الخ ...ثم اختيار اسم الكائن والتسجيل المناسب مثال التالي السابق ....الخ .

١. عند عمل الماكرو يتم اغلاقه وحفظه باسم .

٢. لإدراج الماكرو في النموذج مثلا ......سوف اشرح طريقة بسيطة جدا بدون تعقيد ......افتح نموذجك على التصميم وضع حجمة مصغر جزئيا لكي تظهر قاعدة البيانات معه على نفس الشاشة ثم اضغط ضغط مستمر واسحب الماكرو من قاعدة البيانات للنموذج وقم بتنسيق الازرار .

لعمل مجموعة ماكرو

مجموعة الماكرو هي عدة اجراءات يتم حفظها باسم واحد أي كملف واحد ومن ثم وضعها في الكائن كالنموذج كزر واحد .

مثال: لعمل ماكرو لفتح نموذج وتكبير الشاشة ثم ظهور رسالة ترحيبيه ....يتم اختيار الإجراءات التالية open form ثم maximize ثم msgbox ثم حفظهم جميعا كملف واحد ووضعه كزر واحد في النموذج

ملاحظة

للتعديل بالماكرو من زر تصميم

لعمل قائمة تحتوي على الماكرو

كما سبق وشرحنا إدراج قائمة يمكن من قائمة عرض ثم اشرطة الادوات ثم من تخصيص جهة الاوامر نختار قائمة جديدة ثم نضعها بالضغط والسحب جهة القوائم ومن تعديل التحديد يمكن ان نغير اسم القائمة .
ومن جهة وحدات الماكرو يتم بالضغط والسحب نقل ألما كروات للقائمة الجديدة حيث يمكن تغيير الاسم أو شكل الزر من زر تعديل التحديد .

ربط الجداول

ربط الجاول يعني انشاء علاقة ارتباط بين جدولين أو أكثر .
وتستخدم الحقول المشتركة بين الجداول في عملية الربط ، ويجب أن تكون البيانات الموجودة بين الحقول المشتركة متشابهة .
وتوجد ٣ أنواع من العلاقات هي :

١- علاقة ارتباط رأس برأس (واحد مقابل واحد)

٢- علاقة ارتباط رأس بأطراف (واحد مقابل مجموعة)

٣- علاقة ارتباط أطراف بأطراف (مجموعة مقابل مجموعة)

ومن شروط إنشاء العلاقة بين جدولين :

١. تأكد أن كلا الجدولين المراد إنشاء علاقة بينهما يشتملا على حقل أو حقول متشابهة في كل شيء .

٢. يجب أن تعرف من سيكون الجدول ارئيس (Primary) ومن سيكون الجدول التابع (Related) . بحيث أنه إذا لم يكن الجدول التابع يشتمل على حقل يتطابق مع حقل المفتاح الأساسي في الجدول الرئيسي قم بإضافة حقلاً جديداً في الجدول التابع وبنفس مواصفات حقل المفتاح الأساسي في الجدول الرئيسي .

يقوم البرنامج بتحديد نوع العلاقة وفقاً لخصائص الحقول المستخدمة . 20

انشاء العلاقات بين الجداول :

١. افتح نافذة قاعدة البيانات .

٢. اختر الأمر علاقات من قائمة أدوات أو انقر زر علاقات من شريط الأدوات فتظهر نافذة علاقات كما في الشكل .



٣. إذا لم يظهر مربع (إظهار جدول) تلقائياً اختر أمر إظهار جدول من قائمة علاقات أو انقر زر **** من شريط الأدوات .

٤. من مربع إظهار جدول قم باختيار الجداول التي تريدها وقم بإضافتها بواسطة زر إضافة ، ثم انقر زر الإغلاق فتظهر نافذة العلاقات محتوية على الجدول التي تم اختيارها .

٥. من نافذة علاقات اسحب الحقل من الجدول الذي تريد ربطه مع الجدول الآخر حيث يعتبر هذا الجدول هو الجدول الأساسي ، ضع الحقل الذي سحبته فوق حقل مشابه له في الجدول الآخر (الجدول المرتبط) .

٦. يظهر مربع حواري بعنوان (تحرير علاقات).
تأكد أن الحقل المشترك في الجدولين ظاهراً في كلا الجدولين ، وإلا انقر السهم الموجود على يسار اسم الحقل ثم اختر الحقل الصحيح من القائمة المنسدلة .

٧. انقر زر (إنشاء) من المربع الحواري (تحرير علاقات) فيتم غلق المربع الحواري ويظهر خط يصل الحقلين المتشابهين في كلا الجدولين ، ليوضح أن علاقة ارتباط قد أنشئت بين الجدولين .

٨. قم بحفظ وإغلاق النافذة .

٩. عند ربط أكثر من جدول تظهر رموز على الخطوط الواصلة بين جدولين دلالة على ارتباطهما ، رمز الدائرتين معناة أن أكثر من سجل في هذا الجدول مرتبطة بسجل واحد في الجدول الآخر ، ومعنى الرقم ١ أن كل سجل من هذا الجدول يقابله سجل في الجدول الآخر .

٢١ ولإلغاء العلاقة بين جدولين ، انقر الخط الواصل بين الجدولين ثم اضغط مفتاح DEL .

الوحدات النمطية

لغات البرمجة المستخدمة في برنامج الاكسس هيvisual basic Sql Macro

الوحدة النمطية هي برمجة بلغة access basic وهي لغة visual basic مع ربطها في برنامج الاكسس

لعمل وحدة نمطية

١-من وحدة نمطية اختر جديد يظهر لك نافذة للبرمجة

٢-اكتب بها التالي:-

Function function name()

حيث function name هو اسم التطبيق المستخدم وهو اختياري

وعند اخذ enter يظهر أمر end function تلقائيا .

٣-ثم اكتب الأوامر المطلوبة بين أمر function و end function :-

ملاحظة أوامر الاكسس بيسك هي نفسها أوامر البيسك ....فإذا كنت مهتم بها فابدأ بدراسة أوامر الفيجوال بيسك واليك بعض منها :-

أمر كتابة رسالة مثلا :

Msgbox"الرسالة "

أمر إنشاء مربع إدخال

Inputbox("الرسالة المطلوبة ")

٤-احفظ الملف باسم واخرج منة .

لإدراج الوحدة النمطية للنموذج او التقرير كالتالي:-

١-افتح النموذج أو التقرير على التصميم

٢ –انشأ زر أمر بحجم مناسب (إذا ظهر لك المعالج الخاص به اغلقة)

٣-اضغط بيمين الماوس على زر الأمر ثم اختر خصائص ثم باب حدث ثم اختر الامر عند النقر على الماوس ثم

٤-اكتب الأمر :-

=اسم التطبيق()

اسم التطبيق الذي تم وضعه سابقا وليس اسم الملف

٥-ابدأ التنفيذ.

مثال :-

المطلوب عمل مربع ادخال بحيث عند الضغط على زر الامر يظهر هذا المربع والذي يحتوي على الجملة " ادخل كلمة السر من فضلك "

١-يتم فتح وحدة نمطية جديدة

٢-يتم كتابة الأمر

Functionاسم التطبيق()

يتم كتابة الأوامر التالية

Dimالمتغير as string

المتغير =inputbox("ادخل كلمة السر من فضلك ")

يتم حفظ الوحدة النمطية باسم

يتم فتح النموذج المراد الوضع به وادراج زر امر وبيمين الماوس على زر الامر وعند امر عند الضغط على الماوس يتم كتابة

=اسم التطبيق()

مهارات

(١)عرض بيانات محددة في نموذج آخر

غالباً مانحتاج لفتح نموذج لعرض بيانات محددة مثلا لو كنا نعرض في النموذج الأول بيانات فلان ثم نريد أن نفتح النموذج الثاني على بيانات أخرى لفلان نفسه فما هي الطريقة لربط البيانات بين النموذجين ؟



هذا الشرح يجيب على هذا التساؤل .

طبعاً –وهي القاعدة الأساسية في الموضوع- أنه لابد أن يكون هناك رابطة بين الجدولين المبني عليهما النموذجين وإلا كيف يمكن عرض البيانات .

والرابط هو أن يكون بين جدولي النموذجين علاقة أو هما جدول واحد ولكن مقسوم على النموذجين وتتضح المسألة أكثر فيما يأتي .

لنفرض أن لدينا نموذجين الأول فيه بيانات أشخاص وفي الثاني أرقام الهواتف لكل منهم ونريد عندما نختار اسم من قائمة على النموذج الأول ، أن يفتح النموذج الثاني الذي فيه أرقام الهاتف على السجل المطابق (سأكمل الشرح على المثال المرفق في الأسفل) .

خطوات الإنشاء :

١- أنشئ قائمة منسدلة –لاتنسى تفعيل المعالج الموجود على شريط الأدوات :



عند إنشاء القائمة يشتغل المعالج وتظهر الشاشة التالية : 22

معالج مربع تحرير وسرد

ينشئ المعالج مربع تحرير وسرد، الذي يعرض قائمة قيم يمكن الاختيار منها. ما هي الطريقة التي تريد بها مربع تحرير وسرد الحصول على القيم؟

- (•) أريد أن يبحث مربع تحرير وسرد عن القيم في جدول أو استعلام.
- ( ) سوف أكتب القيم التي أريدها.
- ( ) ابحث عن سجل في النموذج يستند إلى القيمة المحددة في مربع تحرير وسرد.

إلغاء الأمر | < السابق | التالي > | إنهاء

23 **: اتركه على الاختيار الأول ثم اضغط التالي فتظهر الشاشة التالية**

معالج مربع تحرير وسرد

ما هو الجدول أو الاستعلام الذي يجب أن يوفر القيم لمربع تحرير وسرد الخاص بك؟

الجدول: جدول الأسماء

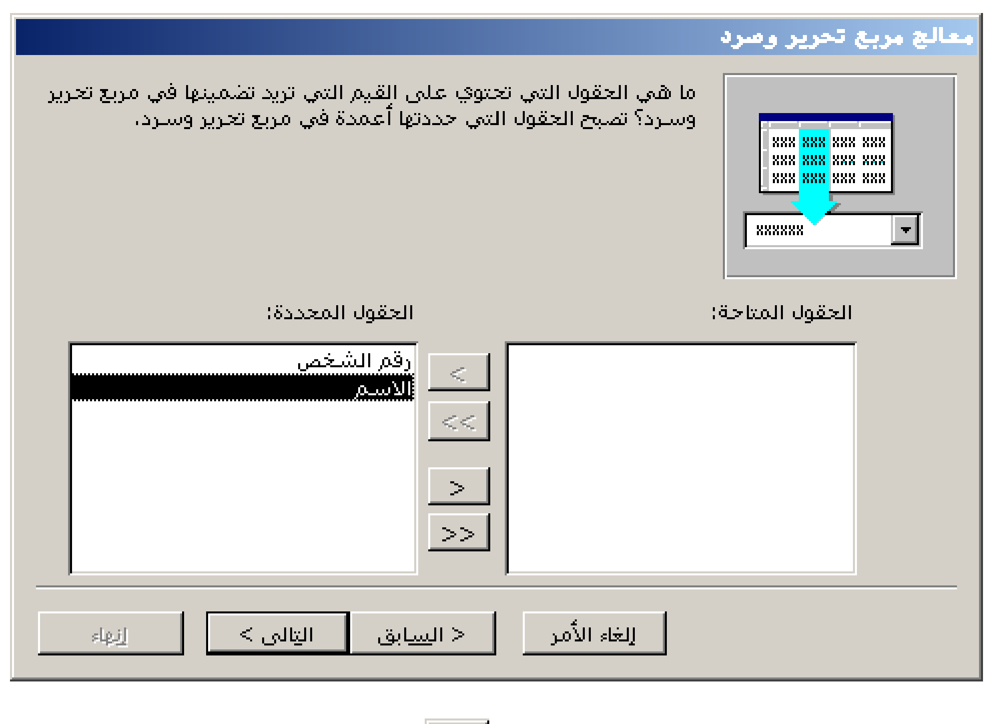
الجدول: جدول الهواتف

عرض

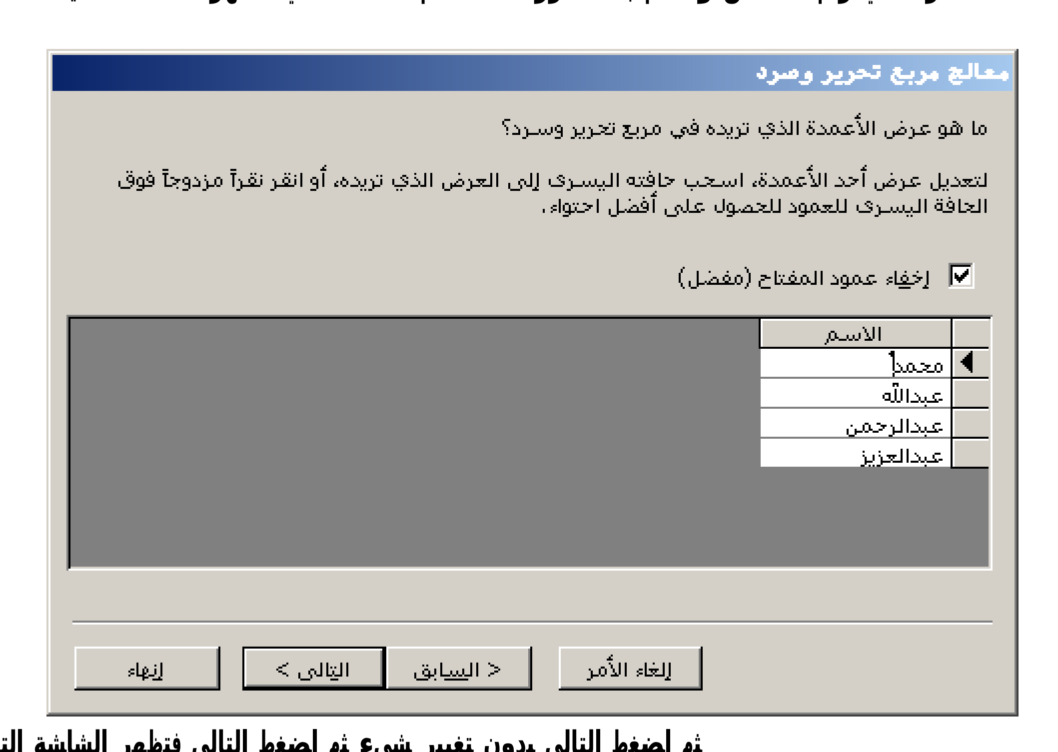
(•) جداول ( ) استعلامات ( ) كلاهما

إلغاء الأمر | < السابق | التالي > | إنهاء

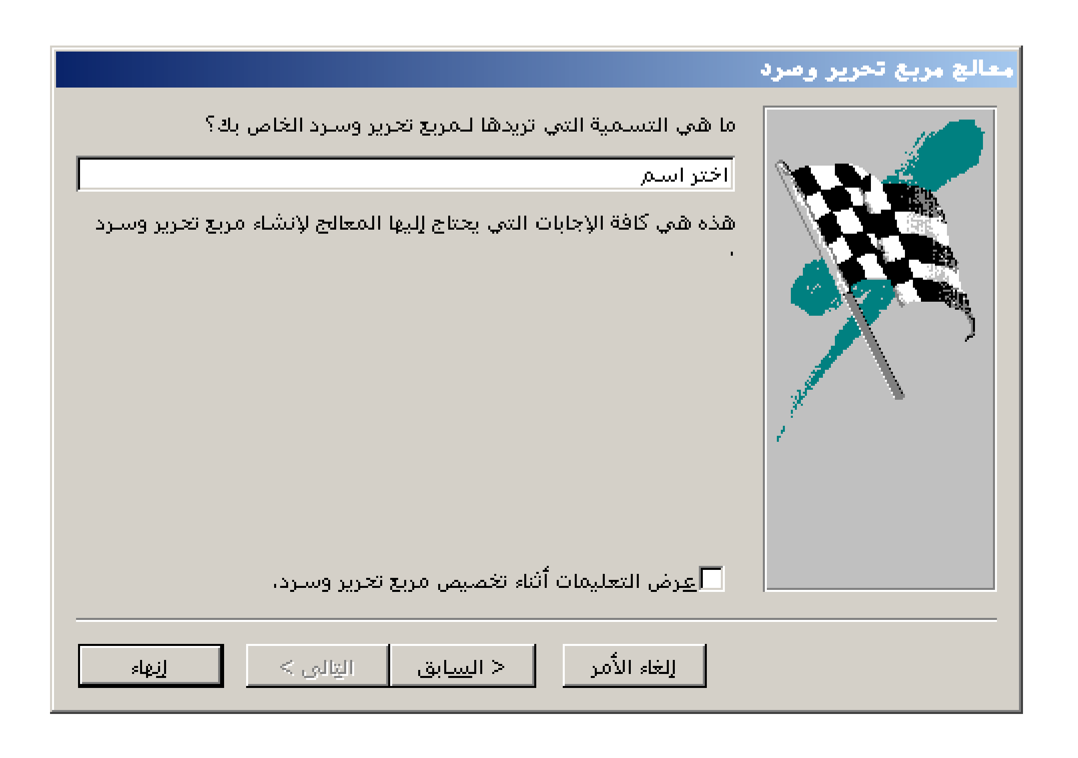
٤- اختر جدول الهواتف ثم اضغط التالي فتظهر الشاشة التالية :



٥- اختر حقلي رقم الشخص والاسم بضغط زر >> ثم اضغط التالي فتظهر الشاشة التالية :



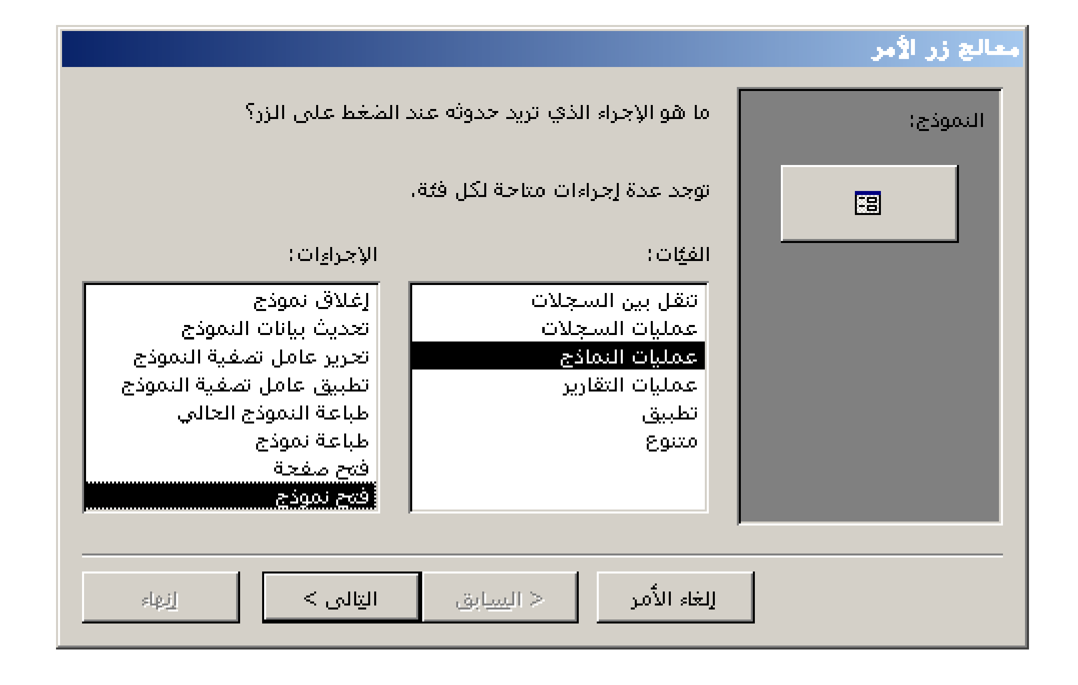
ثم اضغط التالي بدون تغيير شيء ثم اضغط التالي فتظهر الشاشة التالية : 24



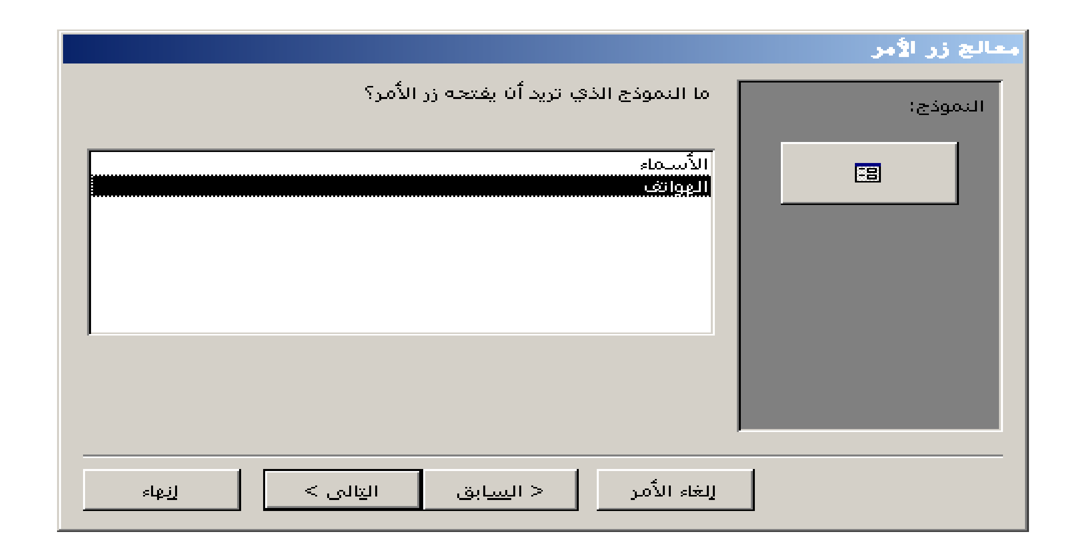
اكتب اختر اسم ثم اضغط إنهاء .

٧- اظهر خصائص القائمة ثم غير اسمها إلى اختر_اسم

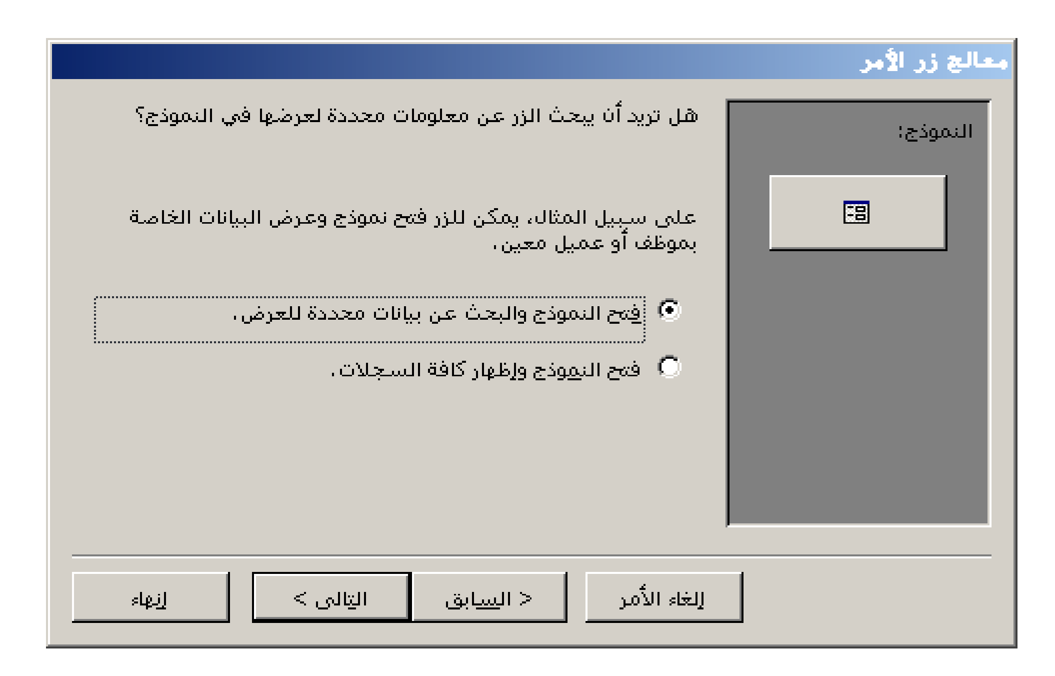
٨- أنشئ زر أمر على النموذج فتظهر الشاشة التالية :



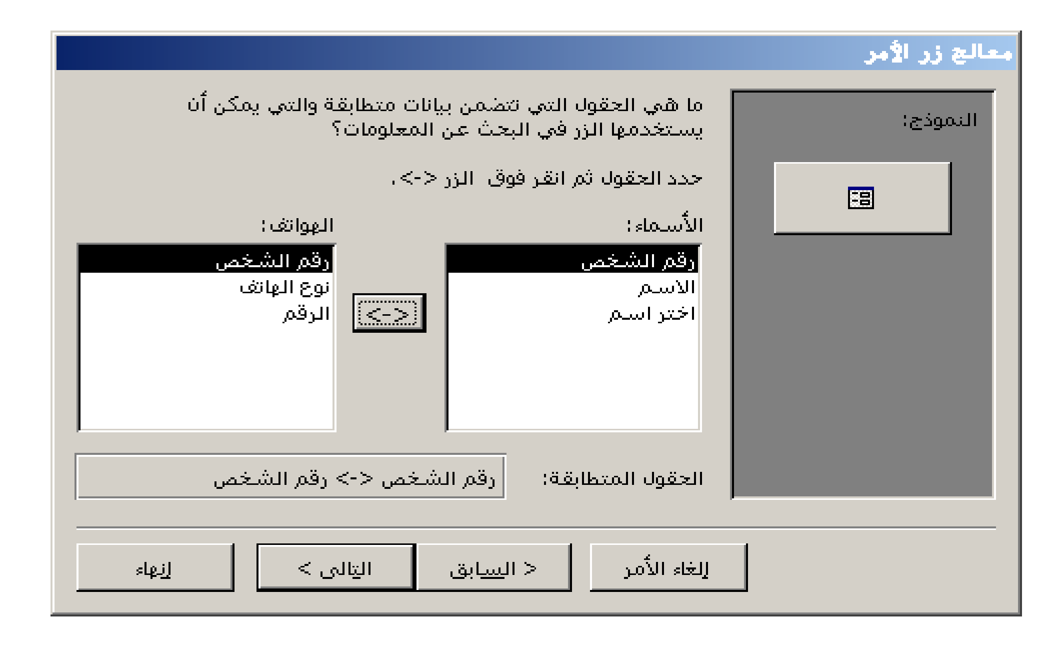
٩- اختر عمليات النماذج ومن اليسار اختر فتح نموذج ثم اضغط التالي فتظهر الشاشة التالية :



١٠- اختر جدول الهواتف ثم اضغط التالي فتظهر الشاشة التالية :



١١- اختر الخيار الأول : فتح النموذج والبحث عن بيانات محددة للعرض فتظهر الشاشة التالية :



١٢- اختر رقم الشخص من اليمين ورقم الشخص من اليسار بالنقر على كل منهما مرة واحدة ثم اضغط هذا الزر  فتظهر أمام كلمة الحقول المتطابقة في الأسفل رقم الشخص <-> رقم الشخص ، ثم اضغط التالي ولاتغير شيء ثم اضغط التالي مرة أخرى فتظهر الشاشة التالية :

معالج زر الأمر

النموذج:

ما هو اسم الزر الذي تريده؟

يساعدك الاسم الذي له معنى في الرجوع إلى الزر لاحقاً.

فتح_نموذج_الهاتف

هذه هي كافة المعلومات التي يحتاج إليها المعالج لإنشاء زر الأمر.

عرض تعليمات عند تخصيص الزر.

إلغاء الأمر | < السابق | التالي > | إنهاء

١٣- اكتب الاسم فتح_نموذج_الهاتف ثم اضغط زر إنهاء .

١٤- الآن لو جربت فتح النموذج سيفتح على السجل المعروضة بياناته ، ولكن نحن نريده أن يفتح نموذج الهواتف حسب الاختيار من القائمة وليس حسب السجل الحالي ، لذلك نحتاج للتعديل في كود زر الأمر الذي أنشأه المعالج .

١٥- أظهر صفحة الخصائص لزر الأمر ثم انتقل لصفحة التبويب حدث وفي حدث عند النقر انقر الزر ذو الثلاث نقط [...] وستظهر الوحدة النمطية الخاصة بالنموذج وفيها الكود التالي :

```
On Error GoTo Err_فتح_نموذج_الهاتف_Click

Dim stDocName As String
Dim stLinkCriteria As String

stDocName = ChrW(1575) & ChrW(1604) & ChrW(1607) & ChrW(1608) & ChrW(1575) & ChrW(1578) & ChrW(1601)

stLinkCriteria = "[رقم الشخص]=" & Me![رقم الشخص]
DoCmd.OpenForm stDocName, , , stLinkCriteria

Exit_فتح_نموذج_الهاتف_Click:
Exit Sub

Err_فتح_نموذج_الهاتف_Click:
MsgBox Err.Description
```

```
Resume Exit_فتح_نموذج_الهاتف_Click
```

والسطر المطلوب تغييره هو :

```
stLinkCriteria = "[رقم الشخص]=" & Me![رقم الشخص]
```

وكما تلاحظ يشير الكود لمربع النص المسمى رقم الشخص والمطلوب أن يشير إلى القائمة المنسدلة التي أنشأناها منذ قليل وهي باسم اختر اسم ، لذلك غير السطر السابق بالسطر التالي :

```
stLinkCriteria = "[رقم الشخص]=" & Me![اختر اسم]
```

في هذه الأسطر سأتكلم عن الفرق بين الأحداث في الوحدات النمطية والأحداث في الخصائص لأن بعض المبتدئين يلخبطون بينهم ، والأمر الآخر حتى يستفاد منها أكثر .

إذا قيل –وهذي خذوها قاعدة – ضع هذا الكود أو هذا السطر في حدث كذا وكذا فالمقصود أن تضع ذلك الكود أو السطر في الوحدة النمطية للنموذج في الحدث المذكور وليس أن تضعه في الخصائص .

ولمعرفة الفرق بينهما افتح النموذج في عرض التصميم أظهر صفحة الخصائص ثم أظهر صفحة التبويب حدث الآن الأحداث الظاهرة يمكن أن تضع فيها استدعاء لدالة عامة أو إجراء عام أو ماكرو ولكن لايصح أن تضع فيها اسطر كود فهذا غلط بالتأكيد .

إذن كيف نضع الأسطر في الوحدة النمطية الخاصة بالنموذج ؟

جواب : لنفرض أنك تريد تكبير النموذج للحد الأقصى عند فتحه وتريد أن تضع هذا السطر في حدث عند الفتح :

```
DoCmd.Maximize
```

فعند وضع المؤشر داخل المربع المجاور لكلمة عند الفتح يظهر زرين على اليسار الأول عليه سهم رأسه للأسفل والثاني عليه ثلاث نقط .

الأول اللي عليه السهم لسدل قائمة للإختيار منها كما في الصورة التالية :



الذي يظهر في القائمة هي عبارة :

[إجراء حدث]

ثم تحتها كل الماكروات في القاعدة .

طبعا هنا نختار إما إجراء حدث وهنا سيكون الحدث في الوحدة النمطية الخاصة بالنموذج أو ماكرو وبقي الأمر الثالث وهو استدعاء دالة عامة وإليك شرح لهذه الطرق .

أولا : أجراء حدث

عند إختيارنا لـ [إجراء حدث] فهذا يعني أن الإجراء سيكون في الوحدة النمطية التابعة للنموذج ، وبهذه المناسبة الوحدات النمطية بالنسبة لمكانها تنقسم إلى قسمين :

١- الوحدة النمطية العامة وهي التي تظهر في قسم وحدات نمطية في إطار قاعدة البيانات .

٢- الوحدات النمطية الخاصة بالنماذج أو التقارير وهي لاتظهر في إطار قاعدة البيانات ولكن يمكن الوصول إليها عن طريق نقر قائمة عرض ثم تعليمات برمجية .

بعد اختيارك لـ [إجراء حدث] من القائمة انقر المربع الأيسر ذو الثلاث نقط ، وسيظهر إطار محرر الوحدات النمطية لـ برامج الأوفس .

في مثالنا السابق بعد فتح إطار محرر الوحدات النمطية سيظهر المؤشر بين سطرين كالتالي :

```
Private Sub Form_Open(Cancel As Integer)

End Sub
```

وسيكون المؤشر بين هذين السطرين لكتابة أسطر الكود ، نضع كما في مثال المذكور سطر الكود ليكون كالتالي :

```
Private Sub Form_Open(Cancel As Integer)
DoCmd.Maximize
End Sub
```

الآن الكود جاهز للعمل .

ثانياً : تشغيل ماكرو

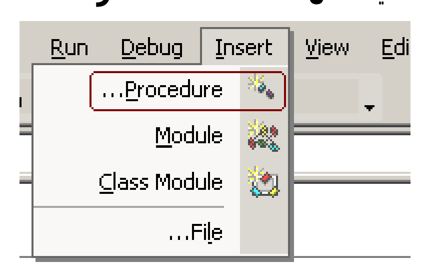
لتشغيل ماكرو لأداء نفس العمل ، أنشئ ماكرو جديد وفي عمود إجراء اختر Maximize ثم احفظ الماكرو باسم تكبير مثلاً ، والآن في الخصائص في حدث عند الفتح للنموذج اسدل القائمة تجد هذا الماكرو المسمى تكبير اختره من القائمة .

ثالثاً : استدعاء دالة عامة أو إجراء عام

يمكنك أن تستدعي دالة أو إجراء عام بطريقة تشبه طريقة الماكرو .

أولاً: افتح أي وحدة نمطية عامة أو أنشئ جديدة .

ثانياً : من قائمة Insert اختر Procedure انظر الصورة :



وعند ظهور مربع الحوار :



اكتب في الاسم max مثلاً أو أي اسم تحب ثم اختر في النوع Type دالة Function وفي الأسفل في المجال Scope اتركه على عام Public .

الآن سيظهر الكود بالشكل التالي :

```
Public Function max()
DoCmd.Maximize
End Function
```

ثالثاً : افتح النموذج في عرض التصميم وأظهر الخصائص وفي صفحة التبويب أحداث في حدث عند الفتح اكتب :

```
=max
```

ونفس الطريقة تستخدم مع الإجراء العام فقط اختر في مربع الحوار السابق Sub بدلاً من Function .

للمزيد عن الإجراءات والدوال انظر الوحدات النمطية .

(٣) لنسخ جدول مثلا من برنامج الاكسس لبرنامج الوورد

1. يتم فتح الجدول
2. حيث ينسخ الملف لبرنامج الوورد ms word ثم نختار النشر باستخدام office من قائمة أدوات نختار ارتباطات rich text format . ((rtf وحفظه بنفس الاسم الاصلي ولكن بامتداد

٢٥ جدول مثلا لبرنامج الأكسل

1. يتم فتح الجدول
2. excel حيث يتم نسخ الجدول لبرنامج ms excel ثم نختار التحليل باستخدام officeثم من قائمة أدوات نختار ارتباطات وحفظه بنفس الاسم وبنفس امتداد برنامج الاكس xls .

(٥) ضغط قاعدة البيانات لنقلها لدسك مرن

تأكد أولا من حجم القاعدة من قائمة ملف ثم خصائص قاعدة البيانات ثم من باب عام .

اذا كانت القاعدة اكبر من حجم الدسك فيجب ضغطها أولا ثم انسخها للدسك المرن كالتالي :

١ - يتم إغلاق القاعدة

٢-ثم من قائمة أدوات اختر ادوات مساعدة لقواعد البيانات ثم اختر ضغط قاعدة البيانات ونحتار القاعدة المراد ضغطها ثم الضغط على زرضغط .

٣-ثم نختار الجهة المراد النسخ لها مثل الدسك المرن a ثم اضغط زر حفظ .

(٦)لاخفاء قاعدة بيانات

1. من قائمة إطار نختار اخفاء
2. ولإظهارها من قائمة إطار ثم اظهار .

ملحق عن لغة الXml

وماذا عن لغة XML؟

إنّ XML هي طريقة لتمثيل أيّ بيانات منظّمة، وذلك بتحويلها لنصّ يعبّر عنها.. وبهذا تصلح لغة XML للتعبير عن أيّ نوع من أنواع البيانات، مثل الجداول والصور وغيرهما.

ورغم أنّ الملفّات النصّيّة أكبر حجما من الملفّات الثنائيّة Binary Files، إلا إنّ الأولى صالحة للتعامل مع أيّ تطبيق بل مع أيّ نظام تشغيل.. لهذا فقد صارت لغة XML في السنوات الأخيرة هي أنسب وسيلة لنقل البيانات عبر الإنترنت، وذلك حتّى تتجاوز مشاكل عدم التوافق بين التطبيقات وأنظمة التشغيل المختلفة.

متطلبات العمل مع XML .

جميع ما تحتاجه إلى استخدام XML موجود في معظم الحواسب .

- برنامج تحرير نصوص عادى مثل المفكرة (Notepad) .

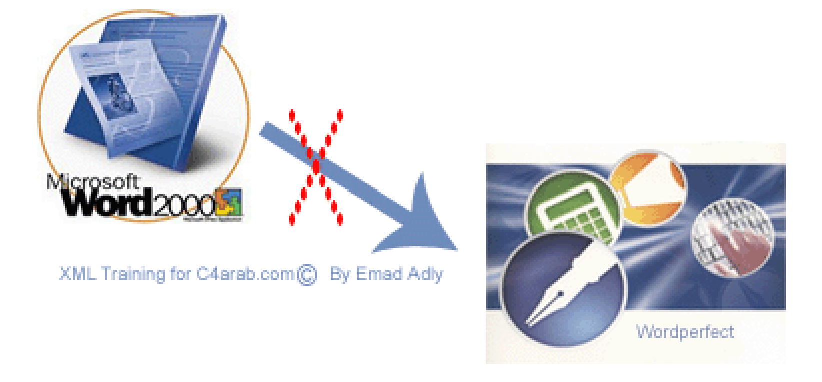
- متصفح يمكنه استعراض مستندات XML مثل Internet Explorer 5.5 أو احدث .

قبل التعرف على تقنيات XML والدافع إلى تعلمها نلقى نظرة على هيئة البيانات والملفات والنصوص ولغات الترميز الأخرى وتاريخها .

فلكي نفهم لغة الترميز يجب علينا فهم كيفية حفظ البيانات وكيفية الوصول لها . فهناك نوعان رئيسان من الملفات التي يفهمها الحاسب .الملفات النصية والملفات الثنائية.

أولا : الملفات الثنائية :-

وهى عبارة عن سلسلة من البتات (٠-١) صفر و واحد ويتم التعرف عليها وفهمها بواسطة التطبيقات التي أنشأتها . ونلاحظ ذلك في عدم القدرة على قراءة ملف ما إلا بواسطة التطبيق الذي صنع لأجله فمثلا لو أرت فتح مستند مكتوب بواسطة برنامج معالج كلمات ما ولنفرض Word Perfect على برنامج Microsoft Word فان الملف لم يفتح أو أنة يفتح بشكل غير مناسب وذلك لأنة لم يعد للعمل على هذا التطبيق بالاختلاف انه صمم لكي يناسب العمل على تطبيق أخر . ولحسن الحظ فان معظم تطبيقات معالجة الكلمات اليوم تحتوى على محولات تستطيع فتح وقراءة الملفات التي تم إنشاءه على معالجات أخرى .



أن الميزة الجيدة للملفات الثنائية هي سهولة فهم الشفرات الثنائية من قبل الحاسب ، بما أن بنية الحاسب هي أصلا بنية ثنائية تعتمد على الوحدات والأصفار فان قراءة الملفات الثنائية ستكون أسرع من قراءة هيئات الملفات الأخرى .

وكما ذكرنا السيئة الوحيدة هي عدم القدرة على فتح تطبيق تم بناءة على تطبيق أخر . وقد يصل الأمر إلى عدم إمكانية فتح وقراءة الملف في نفس التطبيق ولكن ضمن منصة تشغيل Platform مختلفة أو ضمن إصدارة سابقة لنفس التطبيق.

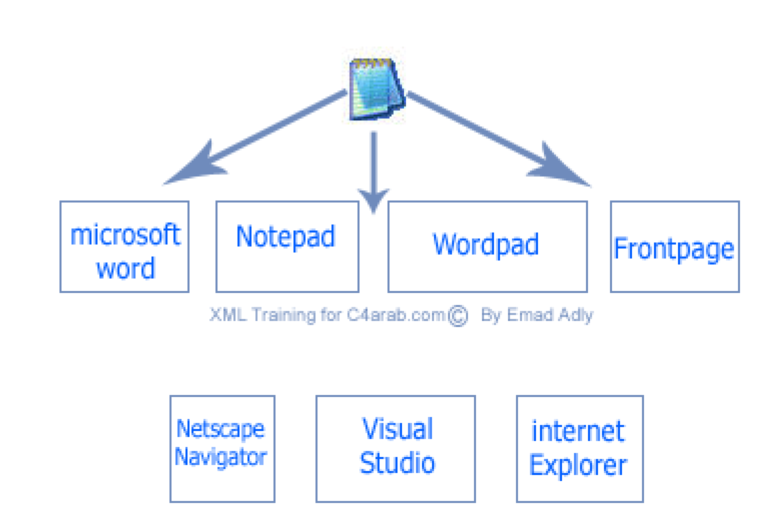
ثانيا : الملفات النصية -:

الملفات النصية تشبه أيضا الملفات الثنائية . فهيا أيضا تمثل سلسلة من البتات (٠ - ١) صفر و واحد ولكنها تعمل مع بعضها بصورة قياسية بحيث تشكل أرقاما على سبيل المثال .

١١٠٠٠٠٠١

هذه المجموعة من البتات تترجم بحيث تمثل الرقم ٩٧ والذي يتم أيضا بحيث يمثل الحرف (a) .

وينتج عن هذه الشفرات القياسية أن الملفات النصية يمكن قراءتها من قبل معظم التطبيقات هذا بالإضافة إلى إمكانية قراءتها من قبل البشر .



إذا قمنا بكتابة ملف نصي فان إي شخص في العالم يستطيع فهم لغة النص ويمكن قراءته بأي محرر نصوص يفضله . فهو يساعد على مشاركة وتبادل المعلومات مع الآخرين .

السيئة الوحيد لهذا النوع من الملفات هو أننا لا نستطيع تهيئة النص المكتوب بالطريقة التي نرغب بها كما في الملفات الثنائية فلا نستطيع توسيط النص مثلا أو تلوينه بلون معين أو نجعل النص مائلا .

تاريخ لغات الترميز :-

لقد لاحظنا ميزات الملفات الثنائية ، فهي سهلة الفهم بالنسبة للحاسب ولاحظنا أيضا ميزات الملفات النصية فهي قابله للتشارك وتبادل المعلومات بشكل واسع أليست صفقة رابحة لنا أن توجد هناك هيئة ملفات تجمع بين قابلية تبادل المعلومات الواسعة التي توفرها الملفات النصية بالإضافة لفعالية وإمكانيات التخزين القوية التي توفرها الملفات الثنائية .

أن هذه الفكرة ليست جديده . فمنذ أن توفرا الحاسب أما المبرمجين وهم يبحثون عن طرق لتبادل المعلومات بين مختلف الأنظمة والبرامج وكخطوة أولية لذلك ظهرت لغة الترميز المعممة القياسية SGML وهى اختصارا إلى

Standard Generalized Markup Languge كانت تلك عبارة عن لغة نصية تستخدم لترميز البيانات . مثل الملفات الثنائية ولكن بطريقة تشرح نفسها بنفسها وهو ما يعرف بالملفات ذاتية الوصف Self Describing وهذه اللغة احتلت موضعا مرموقا في العديد من أنظمة إدارة البيانات الكبيرة . فهيا تحتاج إلى الكثير من الاعتبارات عن ترميز البيانات الضخمة ونتيجة إلى ذلك أصبحت لغة معقدة جدا ولكن من التعقيد تأتى القوة .

مقدمة إلى تقنية XML وما حولها

لغة الترميز الموسعة eXtensible Markup Language التي يرمز لها بالاختصار XML وهى تستخدم في وصف وتخزين وتنظيم البيانات بخلاف لغة HTML التي تستخدم لكيفية عرض البيانات على المتصفح .

تحدثنا سابقا عن لغة SGML وذكرنا إنها لغة معقدة لا تصلح لتبادل المعلومات عبر الشبكة . وعلى الرغم من أنها لغة لوصف البيانات على المتصفح ولا يمكن استخلاص معلومات معينة حول شخص ما ولسبب قصور لغة HTML في قدرتها على وصف أنواع محددة من المعلومات دعت الحاجة لوجود لغة تحقق ذلك وهى لغة الترميز الموسع أو XML وهى لغة أيضا مشتقة من لغة SGML ومتوافقة بصورة كبيرة مع هذه اللغة. هذا يعنى إن أي مستند يتبع مصطلحات وتعابير لغة XML فأنة يتبع أيضا مصطلحات وتعابير لغةSGML .

وهنا يجب ملاحظة شي هام أن لغة XML ليست لغة في الأصل فا XML تصف مجموعة من التعابير التي تستخدمها لبناء لغاتك الخاصة على سبيل المثال لنفترض أن لدينا بيانات حول اسم شخص ما وانك تريد تبادل هذه البيانات مع الآخرين يمكنك تمثيل هذه البيانات في ملف نصي بالصورة التالية.

Emad Adly

أو بصيغة HTML بهذا الشكل .

```
<html>
<head><title>Name</title></head>
<body>
<p>Emad Adly</p>
</body>
</html>
```

يمكن تمثيل هذه البيانات في XML بالشكل التالي .

```
<name>
    <first>Emad</first>
    <last>Adly</last>
</name>
```

ذكرنا سابقا أن لغة SGML وXML تسميان بالغات ذاتية الوصف لان البيانات يمكنها بسهولة معرفة أن هذه المعلومات تمثل اسم Name لشخص ما وأيضا هناك بيانات تسمى <first>و بيانات أخرى تسمى <last> يجب آن تكون ذات معنى طبعا المعنى يدل على محتوى المعلومة بداخلها .

لو قمنا بحفظ الملف السابق باسم name.xml فيمكنا فتح هذا الملف بواسطة متصفح الإنترنت لديك بشرط أن لا يقل عن ٥,٥ وسوف يظهر بهذا الشكل .

```
- <name>
    <first>Emad</first>
    <last>Adly</last>
  </name>
```

وبالرغم من أن ملف XML السابق لا يحتوى على إي معلومات حول كيفية العرض فان المتصفح قام باستعراض الملف بصورة لطيفة وبتنسيق لونى مختلف وأيضا البنية الشجرية التي فهمها المتصفح وترجمها أيضا وذلك بالنقر على الرمز (-) بجانب البند <name> وهذه الطريقة مفيدة جدا عندما يكون الملف كبير الحجم .

ونلاحظ أننا لم نقم بوصف البيانات لكي تظهر بهذا التنسيق الموجود ولكن هذا ما يقدمه لنا متصفح الإنترنت فلهذا المتصفح ورقة تنسيق Style Sheet افتراضية مبيتة داخلة مما يمكن المتصفح من عرض إي مستند XML وفق ورقة التنسيق هذه .

وبهذا قد عرفنا مقدمة بسيطة إلى هذه اللغة وفى الدرس القادم سوف نتحدث عن الأسباب التي تدفعنا لتعلم هذه اللغة .

والجدير بالذكر أن لغة XML تتطلب منا بعض القوانين المحددة لكتابه وثائق XML قابلة للعرض . مثلها مثل لغة HTML فيها أيضا تتطلب دراية بطريقة كتابه الـ Tags الخاص بها .

معربات لغة XML Parsers :-

إذا اتبعنا القوانين المحددة وفق لغة XML يمكننا الوثوق من سهولة استخلاص المعلومات . يعود ذلك إلى وجود برمجيات تسمى بالمعربات Parsers وظيفتها قراءة عبارات XML واستخلاص المعلومات من تلك العبارات .

وهى تستخدم في برامجنا للتعامل مع مستندات XML .

فلا داعي للقلق حول كيفية استخلاص المعلومات من ملفات XML. فاليوم افضل من الأمس . في الماضي وقبل استخدام هذه المعربات كان يتحتم عليك القيام بالكثير من العمل لبناء قوانين تحكم هذه المعلومات ولكن الآن مع صيغة XML يمكنك فقط إعطاء معرب لغة XML ملفا كما يلي :-

```
<name>
   <first>Emad</first>
   <last>Adly</last>
</name>
```

سيقوم المعرب باستخلاص البيانات من هذه الشفرة وسيخبرنا بأن هناك بندا للبينات باسم وأن المعلومة المرفقة لهذا البند هي Emad ولا يتحتم على معرب XML معرفة أي قوانين حول موضع الاسم من البيانات .

أن الشفرة المكتوبة بلغة XML مهما كان نوعها لا تؤثر في عمل المعرب فان كنت كتبت شفرة XML بالإنجليزية أو بالعربية أو بأية لغة أخرى فجميعها يمكن قراءتها وفهمها بنفس المعرب . وحتى إذا كان الشخص الذي كتبها لا يعرف شي عن هذه اللغة .

لغة XML لغة مرنة بدرجة كبيرة فهي تهدف إلى أن تكون الأساس للغات تبادل المعطيات المختلفة خاصة عبر الإنترنت فهي تجعل من تشارك الملفات والمعلومات على الإنترنت وتبادلها مسالة سهله جدا.

الترميز الموسع Extensible :-

باعتبار أنة يمكننا التحكم بصورة كاملة في تكوين مستندات XML فيمكننا أن نشكل البيانات بالطريقة التي تحلو لنا .

أو قررنا عدم احتياجنا لمرونة اكبر في المثال السابق فنقوم بكتابة الشفرة بالطريقة التالية .

```
<name>Emad Adly</name>
```

أنت حر في طريقة تمثيل البيانات . يتوقف ذلك على الطريقة التي تمكن برامجنا من استخدام هذه البيانات . أما إذا أردت استخدام المرونة المتوافرة xml فيمكنك القيام بذلك . فاختار ما يناسب احتياجاتك .

من هنا جاءت تسمية هذه اللغة بالترميز الموسع أو Extensible فان أي شخص يستطيع ترميز البيانات نفسها بأي طريقة باستخدام هذه اللغة .

ولكن يجب أن نأخذ بالحسبان أنه تكمن الاستفادة الحقيقية من لغة XML عندما يستخدم الأشخاص نفس الهيئة لتمثيل البيانات الشائعة لان ذلك سيسمح بتبادل المعلومات بصورة اكبر واسهل .

هناك فعلا العيد من المشاريع للوصول إلى مفردات قياسية لأنواع البيانات الشائعة الاستخدام . فعلا سبيل المثال لغة الرسوميات الشعاعية القابلة للتوسع Scalable Vector Graphics أو SVG وهى مفردات XML التي تمثل قاعدة أساسية لتمثيل الرسوميات ثنائية البعد .

ولغة MathMLوهى مفردات XML لوصف الرياضيات كقاعدة أساسية لاتصال الآلات ببعضها البعض .

ولغة الترميز الكيميائية Chemical Markup Languageأو CML هي مفردات XML لأداره المعلومات الكيميائية

وأيضا لغة WML وهى لغة ترميز اللاسلكي والمستخدمة في بناء مواقع يمكن تصفحها باستخدام الهواتف الخلوية

وهناك الكثير من المفردات المتخصصة في مجالات أخرى تجعل من كتابة البيانات باستخدام XML خاصة بنا إلا إن استخدام المفردات القياسية يساعدك على زيادة توافقية مستندات XML مع البرمجيات الأخرى.

مما تتكون لغة XML ؟

تعتبر تقنية XML عائلة خاصة بذاتها بعضها ما يزال في قيد التطوير في سنواته الأولى .

ولكل فرد في هذه العائلة موصفاته الخاصة . النسخة XML 1.0 هي القاعدة الأساسية التي تبنى عليها XML فهي تصف التراكيب التي يجب على مستند XML اتباعها وكذلك القوانين التي يجب على معربات XML تطبقها بالإضافة إلى تعاريف أنواع المستندات DTDs.

لغة Xpathوهى أيضا جزء لا يتجزأ من تقنية XML وهى لغة الاستعلام لعنونة الأقسام في مستند XML فهي تمكن التطبيقات من إيجاد معلومات محدده ضمن مستند XML .

تدخل لغة CSSضمن عائلة XML أيضا وذلك في حالات استعراض مستندات XML البسيطة ويمكن العوض عنها باستخدام لغة XSL وذلك في الحالات المعقدة وهى تتضمن تحويلات خاصة تسمى هذه التحويلات XSLT والتي تستخدم لتحويل مستندات XML إلى أنواع مستندات أخرى .بإضافة إلى أسلوب عرض المعلومات .

الأختان XLink و XPointer هما لغتان تستخدما لربط مستندات XML مع بعضهما البعض بصورة مشابها للوصلات التشعبيه في مستندات HTML .

نموذج كائن المستند Document Object Model أو DOM هذا النوع من عائلة XML هو معروف جيداً لمن تعامل من قبل مع لغة DHTML و JavaScript ألم تمر عليك جملة Document.write قبل ذلك .

فهذا الكائن يمكنه ربط مستندات XML مع لغات برمجة أخرى مع إمكانية الإضافة والحذف التعديل داخل مستندات XML بواسطة لغتك المفضلة

لماذا XML وفيما تستخدم؟

تقنية XML هي تقنية حديثة وفى دور التطوير حاليا وهى كما عرضنها سابقا في هذه السلسلة وعرفنا ما هي XML .سوف نوضح هنا ما الذي تقدمه لنا هذه اللغة .

تعرفنا سابقا أن لبناء مستندات XML يجب علينا معرفة أشياء وتعليمات كثيرة . فلماذا نستخدم كل هذه الأمور . أليس من السهل وضع بعض القوانين لمثال مستند الأسماء السابق شرحه في الدرس السابق . ونضع بعض المعايير لذلك فمثلا يمكننا القول أن الاسم الأول يبدأ عند بداية الملف ، والاسم الأخير يأتي بعد الفراغ الأول . بهذه الطريقة يمكن لتطبيقنا أن يقرا بيانات الاسم الأول بصورة منفصلة عن الاسم الأخير .

ولتوضيح ذلك : لنفترض إننا نريد إضافة الاسم الأوسط لمثالنا فيصبح هكذا .

Emad Adly Faik

وأيضا عند إضافة هذه العملية لا يوجد إي مشكلات . يمكننا تعديل القانون السابق بحيث إن كل شي يأتي بعد الفراغ الأول وقبل الفراغ الثاني يمثل الاسم الأوسط ، وما يأتي بعد الفراغ الثاني يمثل الاسم الأخير .

إلى الآن يمكننا تمثيل البيانات باستخدام هذا القانون وتنفيذه برمجيا ، ولكن ماذا لو قابلنا اسم كالتالي .

Emad Adly Faik Gabala

الآن لدينا اسمين أوسطين . هنا يصبح القانون اكثر تعقيدا وعلما إن البشر تستطيع تميز الاسم الأوسط من الاسم الأول من الاسم الأخير بسهولة فانه من الصعب برمجة ذلك بطريقة منطقية بحيث يمكن الحاسب من تميز ذلك بسهولة .

للأسف فان معظم المبرمجين عندما يواجهون مشكله من هذا النوع يتصرفون لوضع قوانين صارمة بدلا من التعامل مع البيانات المعقدة كما هي . فالمشكلة الأسماء كما سبق يمكن لمطوري البرامج أن يقرروا أن للشخص الواحد اسم أوسط واحدا فقط ولا يمكن للتطبيق قبول أكثر من ذلك .

فالمثال السابق ليس صعب التطبيق ولكن هو يسلط الضوء حول الأسباب الجوهرية التي أدت لظهور لغة XML.
فالمبرمجين يستطيعون تنظيم بياناتهم بعدد غير محدد من الطرق . وفى كل طريقة من الطرق نظام معين لاستخلاص المعلومات التي نحتاجها باستخدام لغة XML

إلى هنا قد ذكرنا مثال بسيط جدا لتوضيح ابسط الأشياء التي تدفعنا كمطورين إلى الدخول إلى عالم XML .

وحتى إن كان هذا المثال غير مفهوم للمرة الأولي فلا تقلق سيمكنك التميز والتوسع اكثر عند البدء في تشغيل وكتابة مستندات XML.

كما يمكن للغة XML العمل على إي منصة تشغيل فهيا ليست مخصصه للعمل عن نظام عين مثل في Visual Basic مخصص للعمل على منصات ويندوز . فيمكنك استخدامها على إي نظام تشغيل حالي أو سوف يظهر مستقبلا .

فمن الأشياء الرائعة عند استخدامك لـ XML هو تقليص زمن التحميل على خادمات الشبكة Web Server بالإضافة إلى عمليات إرسال واستقبال كميات ضخمة من المعلومات بين جهاز الزبون والخادم .

وأيضا من الميزات الجميلة في التعامل مع ملفات XML هي إمكانية تحويل ملفات XML إلى ملفات html بسهوله لعرضها بشكل جميل على المتصفح وذلك بواسطة محولات XSLT . أو أن تعرض مباشرة عبر المتصفح بواسطة ورقة التنسيق css كما في المثال الأول .

استدعاء الإجراءات البعيدة . فهيا تستخدم بروتوكول RPC لذلك فهو يسمح للكائنات الموجودة على جهاز ما باستدعاء الكائنات الموجودة على جهاز أخر للقيام بعمل ما .

أما في مجال التجارة الإلكترونية أحد المواضيع الساخنة حاليا في الإنترنت . التي ستظل ساخنة لفترة طويلة .فمعظم الشركات اليوم تقر بضرورة ربط عملائها معها عبر الإنترنت بد من استخدام الطرق التقليدية .

فكلما احتاجت شركة ما لإرسال بياناتها إلى جهة أخرى عبر الإنترنت فان xml هي التقنية الأفضل لذلك باعتبار أن الهدف من وراءها هو تبادل المعلومات والتشارك عليها .

فهناك الكثير والكثير من الأماكن التي تقدم لنا فيها تقنية XML الفائدة التي نرجوها.

منهجية عمل مستندات XML وتكوينها الهرمي

سنناقش اليوم التركيب الهرمي الذي يكون مستندات XML وكيفية بناء البيانات وكيفية تمثيلها في مستند XML

- عندما نكون أمام كمية معلومات ضخمة أو حتى متوسطة الحجم فمن الأفضل تجميع هذه المعلومات التي تنتمي إلى موضوع واحد مع بعضها البعض بدل من تمثيلها كما هي .

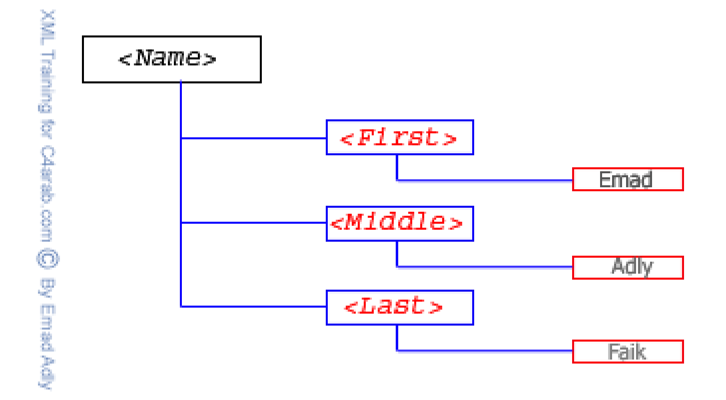
نعطى مثال لكى توضح الصورة .

مثلا هذه المنتدى مجزأ إلى مجموعة من الأقسام والمواضيع الفرعية التي تتبع موضوعات فرعية والتي تتبع بدورها مواضيع رئيسيه بالإضافة إلى تقسيم الموضوع الواحد إلى مجموعة من الفقرات وهى عبارة عن ردود الأعضاء .
فذلك يسهل من عملية استخراج المعلومات .

فنجد مطوري البرمجيات تستخدم هذا النموذج منذ سنوات عدة باستخدام بنية بيانات تسمى **Object model** أو نموذج الكائن وهى مرتبطة مع بعضها بتسلسل هرمي . أيضا في لغة **XML** تجمع البينات في تسلسل هرمي فالبنود في المستند تتبع بعضها البعض بعلاقات **Parent / Child** أو الأب **/** الابن .

وهذه البنود تسمى بالعناصر **elements** وهى أجزاء منفردة من المعلومات .

نأخذ مثال الاسم السابق شرحه ونمثله بطريقة هرمية كالشكل التالي.



نلاحظ أن البند **<Name >** هو أب للبند **<First >** والبند **<First >** هو ابن للبند **<Name >** والبنود **<First>** و**<Middle >** و **<Last >** جميعها انساب لبعضها البعض لان جميعهم أبناء للبند **<Name >**

ونلاحظ أيضا أن النص هو ابن للعنصر الذي ينتمي له فالنص **Emad** يمثل ابنا للبند **<First >**.تسمى هذه البنية من البيانات بالشجرة **Tree** فكل جزى من الشجرة يحتوى على أبناء تسمى بالفروع **Branches** وجميع الأجزاء التي تحتوى على أبناء تسمى بالأوراق **Leaves**

إذا نقول :

Element Content

العنصر **<Name >** يعتبر **element content** لان هناك عناصر تنتمي له وليس مجرد نص فإنه يعتبر محتوى عنصر .

Simple Content

العنصر < First> و < Middle>و< Last> هي محتوى بسيط Simple Content لأنها تحتوى على نص فقط .

Mixed Content

أيضا يمكن للعناصر أن تحتوى على عناصر أخرى وعلى نصوص في تلك الحالة فان للعناصر تلك محتوى مختلط Mixed Content على سبيل المثال .

```
<doc>
    <parent>Computer<em>4</em>arab</parent>
</doc>
```

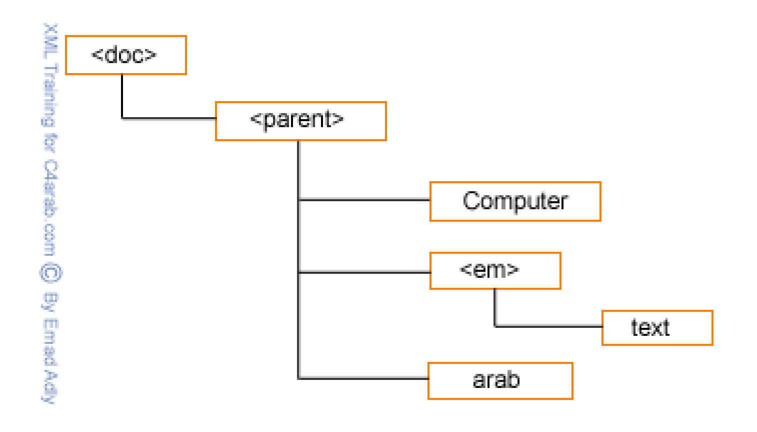
نلاحظ في المثال هذا أن للبند < parent> ثلاث أبناء

نص ،، يحتوى على النص Computer

عنصر ،، < em> وهو عنصر ابن أيضا

نص ،، نص أخر يحتوى على النص arab

فيمكننا الآن تمثيل هذا المثال ببنية شجرية هكذا .



نرجو أن تكونوا تفهمتم كيفية التمثيل الهرمي للبنود والعلامات فيما بينهما فأنك حينما تفهم ذلك ستتمكن من فهم طبيعة لغة XML

منهجية عمل XML وقواعدها :-

١) اللواحق والنصوص والعناصر Tags and Text and Elements

اللاحقة أو ما يطلق عليها البعض والوسم (Tag) هي عبارة عن كلمة أو مصطلح موضوع بين رمزي إحاطة < > يمثل رمزا معرفا لتنسيق ما وذلك في مستندات HTML بينما يمثل اسما لعنصر Elements في مستندات XML

مثل ....

```
<name>
    <first>Adel</first>
    <last>Maher</last>
</name>
```

وكما تلاحظ فان الـ Tag تأخذ الطابع الزوجي فكل لاحقة لها لاحقة مقابله لها تعرف الأولى لاحقة البداية Start Tag وتعرف الثانية بلاحقة النهاية End Tag . الاختلاف بين الاثنين هو أن لاحقة النهاية تحتوى على الرمز "/" .

في XML جميع المعلومات الموجودة بين لاحقة البداية ولاحقة النهاية نسمى بالعناصر Element وبالتالي فان .

<first > هي لاحقة بداية

</first > هي لاحقة نهاية

</first> Adel<firest > هو عنصر

Element Content

النص الواقع بين لاحقة البداية ولاحقة النهاية يسمى بمحتوى العنصر Element content

PCDATA

المحتوى الواقع بين لاحقتين عبارة عن بيانات ويعرف في هذه الحالة ببيانات الرمز المعرب PCDATA وذلك إذا احتوى هذه العنصر على معلومات نصية مثل العنصر <middle > فهوPCDATA

Root Element

المستند ككل بدء باللاحقة <name > وانتهاء باللاحقة</name > فهو يمثل عنصر يحتوى على مجموعة من العناصر وهنا نطلق عليه عنصر الجذرRoot Element

قوانين العناصر

يجب على مستندات XML الخضوع لهذه القوانين كي تشكل فعليا مستندات XML محكمة الهيئة Well-formed XML Documents

- لكل لاحقة بداية لاحقة نهاية مماثلة لها .
- لا يمكن للواحق أن تتداخل .
- يحتوى مستند XML على عنصر جذر واحد فقط.

- لغة XML حساسة لحالة الحروف Case-Sensitive
- لغة XML لا تتجاهل المساحات الفارغة في مستنداتها .

أسماء العناصر :-

لغة XML توفر لك الحرية في تسمية العناصر فهي لا تحتوى على أسماء محجوزة كما في معظم اللغات ، فهي لديها مرونة كبيرة في اختيار الأسماء . ولكن يوجد مجموعة من القوانين التي يجب مراعاتها :-

- يمكن للأسماء أن تبدأ بأحرف لاتينية أو غير لاتينية أو أن تبدأ بالرمز underscore(_) ولكن لا يمكن أن تبدأ برقم أو بعلامة ترقيم .
- بعد الحرف الأول يمكن للأسماء أن تحتوى على أرقام بالإضافة إلى الرمزين " _ " و " . "
- لا يمكن للأسماء أن تحتوى على فراغات .
- لا يمكن للأسماء أن تحتوى على ":" فهو محجوز في XML
- لا يمكن للأسماء أن تبدأ بالأحرف XML سواء كانت بأحرف صغيرة أو كبيرة .
- لا يمكن أن يكون هناك فراغ بين قوس الإحاطة المفتوح < وبين اسم العنصر

٢) الصفات Attributes

أن مستندات XML يمكن أن تتضمن صفات أو سمات معينة attributes الصفات عبارة عن اسم معين تسند له قيمة معينة بحيث يرتبط ذلك الاسم وتلك القيمة بعنصر معين في مستند XML .

مثل ..

```
<name nickname='mrScript'>
    <first>Emad</first>
    <last>Adly</last>
</name>
```

يجب أن تحتوى الصفات على قيم ويجب أن تكون هذه القيم واقعة بين علامتي اقتباس ولا يشترط أن تكون علامة الاقتباس مفردة أو مزدوجة .

يمكن للصفات أن تقدم بيانات وصفية Metadata والتي يمكن أن لا تكون وثيقة الصلة بمعظم التطبيقات التي تتعامل مع المستندات XML

على سبيل المثال إذا علمنا أن بعض التطبيقات يمكن أن تهتم بالاسم المستعار Nickname ولكن معظم التطبيقات لا تهتم بهذه المعلومات فان استخدام هذه المعلومات كصفة سيكون ذا معنى .

إذا ما الذي تقدمه الصفات ولا يمكن للعناصر أن تقدمه .

مثال ..

```
<Name Nicname='MrScript'></Name>
```

أيضا ،،،

```
<Name>
<Nickname>MrScript<Nickname>
</Name>
```

السبب يرجع إلى أن بعض الأشخاص يجدون أن استخدام الصفات تسهل على سبيل المثال فانك لا تحتاج إلى الاهتمام بدرجة تعشيش العناصر ولا تحتاج للقلق حول المعرفات المتداخلة.

فيما أن البعض الآخر يرى إن عملية تداخل وتعشيش العناصر سهل .

مجموعة أخرى ترى أن الصفات تستهلك مساحة اقل بكثير من العناصر . فمثلا المثال الأول لو قمنا بكتابته بواسطة الصفات سيكون بهذا الشكل .

```
<Name Nickname='MrScript' First='Emad' Last='Adly'></Name>
```

ولكن ذلك قلل من مرونة مستندات XML التي تتسم بالمرونة .

- **في النهاية نقول إن عملية أفضلية الصفات عن العناصر أو العناصر عن الصفات تعود على اختيار الشخص وعلى أسلوبه . فاختار منهم ما يشعرك بالارتياح .**

٣) التعليقات
Comments

هي عبارة عن إضافة النصوص التي لا تشكل جزءا من المستن و إنما تخص الشخص الذي يقراء شفرة XML نفسها .

إذ صادف لك وتعاملت من قبل مع أي لغة برمجة فانك تعرف تماما ماذا تعنى بالتعليقات .

وهى تستخدم لوضع بها تعليقات تصف بعض الفقرات و الاكواد لكي تساعدك في فهم شفره مستندات XML

ولكنها ليست مهمة بالدرجة مثلما في إي لغة برمجة أخري حيث أن لغة XML لغة تصف البينات إي أنها ذاتية الوصف تساعدك على فهمها .

تبدأ التعليقات بالرمز < !— وتنتهي بالرمز -- > مثل التعليقات في HTML

مثلا ..

```
<site>
<site-name> الموسوعة العربية </site-name>
<!-- URL Address -->
<site-url>www.c4arab.com</site-url>
</site>
```

٤) العناصر الفارغة Empty Elements

قد توجد في بعض مستندات XML عناصر فارغة من البينات

فمثلا ...

```
<Name>
<first>Adel</first>
<last></last>
</Name>
```

في هذه الحالة تقدم لك XML الحل الأفضل والبديل الذي تكون فيه لا تحتاج إلى معرف نهاية End Tag فتكتب بهذا الشكل

```
< last/>
```

على أن يكون العنصر ثم الرمز "/" بدون لاحقة النهاية المعتادة وهى الحالة الوحيدة التي لا تستخدم فيها لواحق النهايات .

ولو أجريت تجربة بسيطة داخل معرب XML المدمجة بمتصفح إنترنت اكسبلورر ٥,٥ وكتبت الشفرة التالية

أنها سوف تظهر بالشكل التالي

```
- <Name>
    <first>Adel</first>
    <last />
  </Name>
```

٥) تصريح XML Declaration XML

تعد التصاريح في تعريف مستندات XML بأنها تتبع نوعا معينا بالإضافة إلي إعطاء المعرب التعليمات الأخرى

فمثلا هذا التصريح

```
<?xml version ='1.0'  encoding ='UTF-16' standalone = 'yes'?>
```

وهو يجب أن يكون في بداية مستندات XML وليس معنى عدم وجوده يسبب الخطأ ولكن الأفضل تعريف المستندات .

شرح التصريح السابق

- فبدا تصريح XML بالرمز < ؟xml
و ينتهي بالرمز ؟ >

<?

- إذا أضفت تصريح XML يجب أن تضيف أيضا الصفة version أما بالنسبة للصفتين Encoding و Standalonge بهذا الترتيب .

- حاليا فان الإصدار Version يجب أن يكون يحوى القيمة ١,٠ إذا قمت بوضع رقم غير هذا الرقم فان XML المكتوب للتعامل مع مواصفات XML الإصدار ١,٠ سترفض المستند .

- يجب أن يأتي هذا التصريح في أول سطر في مستند XML كما ذكرنا .

<u>الصفة الثانية في تصريح XML هي Encoding</u>

صفة التشفير Encoding Attributes ، لو رجعنا إلى درسنا الأول في هذه السلسلة والذي تحدثنا فيه عن البيانات وأنواعها فلا تستغرب إذا عرفت أن النصوص تخزن في الحاسب على شكل أرقام باعتبار أن الأرقام هي اللغة ألام للحاسب فمثلا الحرف a يمثل في جدول الـ ASCII الرقم ٩٧ والحرف A يمثل الرقم ٦٥ .

هناك نوعان من شفرة ASCII الشفرة القياسية والتي تتكون من سبعة بتات والشفرة الموسعة والتي تتكون من ثمانية بتات أي أن شفرة ASCII الموسعة تستخدم بايتا واحدا لكل رمز وبالتالي فان الشفرة لا يمكن استيعابا اكثر من ٢٥٦ رمز مختلف فهو كافي لتمثيل كل الحروف الأبجدية الإنجليزية الكبيرة والصغيرة بالإضافة إلى العلامات والأرقام ولكن غير كافي لتمثيل لغات أخرى مثل العربية أو اليابانية ... الخ

لهذا السبب وجدت الشفرة الموحدة Unicode .

بعد هذه المقدمة نرجع إلى مثالنا السابق جملة تصريح XML عند الصفة

encoding='UTF-16'

هنا تم ضبط التشفير لمستند XML على انه Unicode وذلك بوضع UTF-16

لذا فقد نصت مواصفات XML على استخدام شفره موحدة عالمية لتمثيل البيانات ولكن للأسف فان القليل يستخدم هذه الشفرة والعديد يستخدم شفرات أخرى مثل ISO-88591 وشفرة windows-1252

ملاحظة إذ لم تقم بتحديد شفرة إي انك لم تقم بوضع هذه الصفة أصلا أن معرب XML يقراء المستند بشفرة UTF-8 أو UTF-16

الصفة الأخيرة في تصريح XML هي صفة Standalone

وهي تعني أن المستند قائم بذاتية فإذا قمت بإضافة هذه الصفة إلى تصريح XML فإنه لهذه الصفة قيمتين yes وno

تشير القيمةyes إلى أن المستند قائم بذاته ولا يعتمد على أي ملفات أخرى

تشير القيمة No إلى أن المستند يمكن إن يكون معتمد أو مرتبطا بملفات أخرى.

٦) رموز نصوص PCDATA غير مسموح بها .

تعرفنا سابقا بالمقصود بالمصطلح PCDATA وعرفنا انه مصطلح مستخدم من قبل SGML يشير إلى البيانات النصية الموجودة داخل العناصر ، فتوجد بعض القوانين تحكم هذه البيانات النصية عند كتابتها لفادى الأخطاء أثناء عمل معرب XML .

فهناك بعض الرموز المحجوزة التي لا تستطيع تضمينها في بينات PCDATA لان تركيب XML يستخدمها من هذه الرموز < والرمز &

فمثلا ..

```
<!-- This is not well-formed XML -->
<comparison> 6 is < 7 & 7 > 6 </comparison>
```

عند تشغيل هذه الشفرة على المتصفح سوف تعرض لك الخطأ بهذا الشكل ..

```
The XML page cannot be displayed

Cannot view XML input using XSL style sheet. Please correct the error
and then click the Refresh button, or try again later.

<comparison> 6 is < 7 & 7 > 6 </comparison>
-------------------^
```

هذا يعني انه عندما يصل المعرب إلى الرمز < سيتوقع أن يجد اسما لمعرف Tag وبدلا من ذلك فقد وجد فراغ Space وحتى إن تجاوز المعرب هذا الخطأ فإنه سيتوقف عند خطا أخر عندما يصل إلى الرمز " & "

فلا تقلق فهناك طريقتان تمكنك من تضمين هذه الرموز ضمن PCDATA إما أن تستخدم الرموز المرادفة Escaping Characters أو أن تستخدم قسم CDATA سنتعرف على الطريقتان الآن في الجزء الثاني هنـــــــا .

الرموز المرادفة Escaping Characters

يمكنك استخدام الرمز < ضمن بيناتك النصية في XML باستبدال هذا الرمز بمرادفة والذي نصت علية XML وهو

```
&lt;
```

ويمكنك أيضا استخدام الرمز & وذلك باستبداله بمرادفة وهو

```
&amp;
```

إذا نقوم بكتابة المستند السابق بصورة سليمة بالشكل التالي ..

```
<comparison> 6 is &lt; 7 &amp; 8 &gt; 6 </comparison>
```

وعند عرضه على المتصفح سيظهر بالشكل التالي ..

```
<comparison>6 is < 7 & 8 > 6</comparison>
```

واليك هذا الجدول يوضح لك الرموز المحجوزة في XML ومرادفاتها .

الرمز المحجوز	الرمز المرادف
&	& amp;
>	& lt;
<	& gt;
‘	'
"	&qu;

توجد العديد من هذه الرموز وهى تعرف باسم Character Referances .

فهي رموز Unicode تبدأ بـ &# و تنتهي بالرمز ; ويتوسطها رموز وفق التمثيل الستعشرى وعلى سبيل المثال يمكنك تمثيل الرمز © في مستند XML بإضافة الرمز البديل ؛ & #١٦٩ أو المرادف الستعشرى له & #xA٩؛

أقسام CDATA

إذا كان لديك الكثير من الرموز التي يتحتم عليك استخدامها رموز بديله لها سيصبح شكل شفرة مستندك مزعجا فيأتي هنا الحاجة إلى استخدام ما يعرف بأقسام CDATA

عند استخدام أقسام CDATA سيقوم المعرب بعدم إعراب النص و إنما التعامل معه على انه عبارة عن بيانات PCDATA بما يحتويها من رموز محجوزة وتكتب أقسام CDATA بهذا الشكل

```
<comparison><![CDATA[6 is < 7 & 7 > 6]]></comparison>
```

أي أن كل ما يوضع بين الرموز < ![CDATA[والرموز]] > أي أن معرب XML سيمرر هذا النص كما هو وسو يتجاهل الرموز المحجوزة.

تظهر قوة التعامل مع أقسام CDATA عندما تريد كتابة شفرة برمجية داخل مستند Xml فمثلا..

```
<script language='javaScript'><![CDATA[
    function myFunc()
    {
        if(0 < 1 && 1 < 2)
            alert("Hello");
    }
 ]]></script>
```

وسيظهر بهذا الشكل على المتصفح ..

```
- <script language="javaScript">
  - <![CDATA[
            function myFunc()
        {
            if(0 < 1 && 1 < 2)
            alert("Hello");
        }
    ]]>
  </script>
```

إعراب مستندات XML :

يرجع السبب لوجود هذه القوانين لكتابة مستندات xml محكمة هو لتسهيل عملية استخراج المعلومات من هذه المستندات .

يسمى معالج لغة xml بمعرب XML Parser ولأنه يقوم ببساطة بأعراب شفرة xml ويوفر للتطبيق المعلومات التى يحتاجها من المستند .

هناك الكثير من المعربات المجانية وسوف اذكر لكم مجموعة منها الآن .

Microsoft Internet Explorer Parser :

أول معربات XML ضمنته شركة مايكروسوفت في متصفحها الإصدار الرابع ولك في بدايات XML ومع الإصدار الخامس من المتصفح زودت لغة xml بأدوات بحيث تم تحديثها إلى الإصدار الأول من هذه المواصفات وعرفت باسم xml 1.0

يمكنك تحميل الإصدار الأخير من معرب xml من موقع مايكروسوفت على هذه الوصلة .

http://msdn.microsoft.com/downloads/webtechnology/xml/msxml.asp

James Clark's Expat :

معرب جايمز كلارك يعد Expat معرب xml 1.0 مكتوب بلغة c وهو أحد معربات XML المجانية ويمكنك تحميلة من الموقع التالي . أخر إصدار هو expat v1.2

ftp://ftp.jclark.com/pub/xml/expat1_2.zip

ولمزيد من المعلومات حول هذا المعرب يمكن مراجعة هذه الوصلة

http://www.jclark.com/xml/expat.html

DataChannel XJ Parser

تعد شركة DataChannel شركة برمجيات حلول الأعمال عملت مع شركة مايكروسوفت لأنتاج معرب xml بلغة جافا يمكنك من الحصول على أخر المعلومات حول هذا المعرب أخر إصدار له من الموقع التالي.

http://xdev.datachannel.com/directory/xml-parser.html

IBM XML4j

تقدم شركة IBM عددا من أدوات وتطبيقات xml بالإضافة إلى المعرب xml4j وهو كتب بلغة الجافا وهو متوفر بالمجان على الموقع التالي

http://www.alphaworks.ibm.com/

Apache Xerces

توفر أيضا مؤسسة Apache للبرمجيات مشروعا أوليا لمعرب XML مازال بإصدارته التجريبية Beta وهو أيضا كتب بلغة الجافا ولغة ++C وباستخدام لغة بيرل أيضا وتجده هنا

http://xml.apache.org/

التعامل مع الأخطاء في XML :

وكما تم تحديد التعامل مع المعلومات داخل مستند XML تم أيضا التعامل مع الأخطاء بواسطة معرب XML فهناك نوعان من الأخطاء وهى الأخطاء Error والأخطاء fatal errors .

فالأخطاء ببساطة هي انتهاك لموصفات وقوانين XML التي تكلمنا عنها هنا حيث يكون الناتج غير معروف .

أما الأخطاء المميتة أو fatal Errors فهي التي تحتم على المعرب عدم الاستمرار في معالجة مستند xml أي إن أي خطأ يجعل مستند xmlغير محكم الهيئة يسمى خطأ مميت .

فهذه الصرامة في كتابة مستندات xml هي التي تزيد من قوتها وليس مجرد تفادى لاخطاء المعربات ولكنها توحيد كتابه مستندات قياسية يمكن التعامل معه من اكثر من متصفح واكثر من منصة تشغيل . بعكس لغة html لا توجد صرامة في كتابة النصوص مما قد تجد متصفحا يقوم بعرض الصفحة بشكل غير لائق في حين متصفح أخر يعرضها كما تريد . وهذا العيب في html وهو سبب من أسباب ظهور لغة XHTML التي تعتبر العوض لقصور html

Html مقابل XML

الاختلاف الرئيسي بين XML وHTML هو أن الـ XML يأخذ وجهة نظر مختلفة عن HTML ، بالرغم من انه ما ذال يستعمل وسم Tags وهو ليس بديل لـ HTML .

XML و HTML صمما كل منهم لهدف مختلف . أن الاختلاف الرئيسي بان XML صمم لوصف تركيب النص وليس ما هو يجب أن يعرض على صفحة المتصفح .باختصار . XML صمم لحمل البيانات ، من الناحية الأخرى HTML صمم لعرض البيانات والتركيز على شكل عرض هذه البيانات . هكذا يمكن أن نقول إن HTML يعمل على عرض المعلومات ، بينما XML يعمل وصف المعلومات .

دعنا نأخذ مثال بسيط :

نقوم بإنشاء ملف file.html ونكتب بداخل هذه الكود

```
<body>

Hello !!
<h1> Welcome To The C4arab.com </h1>
This is normal text. <b> while this is bold text.</b>

</body>
```

وعند عرض هذه الشفرة على المتصفح سوف يظهر لك شي مثل هذا:

Hello !!

Welcome To The C4arab.com

This is normal text. **while this is bold text.**

ويجب أن نلاحظ إن جملة While this is bold text تظهر كنص غامق .

أما إذا قمنا بعرض الملف هذا على انه وثيقة XML بدون تغير في شكل الـ Tags ، فقط قم بتغير امتداد الملف بدل من file.html إلى file.xml

سوف نلاحظ أن متصفحك قام بعرض الملف ولكن بشكل أخرى يشبه هذا الشكل :

```
- <body>
    Hello !!
    <h1>Welcome To The C4arab.com</h1>
    This is normal text.
    <b>while this is bold text.</b>
  </body>
```

نستنتج من المثال السابق أن شفرة HTML التي تحمل الامتداد html تم عرضها بشكل يجب أن يكون معروف أو متوقع عرضها به . وذلك لأنها تحتوى على وصفات Tags تم تعريفها مسبقا لدى المتصفح لكي يمكنه التعرف عليها وعرض البيانات على أساسها . مثل <body> , <h2> , <h1> .. الخ . فأي متصفح يمكنه أيضا إضافة وصفات جديده لعرض البيانات على سبيل المثال متصفح النتسكيب له tags خاص هي <BLINK> ولكن ليس هناك طريقة قياسية لتقديم أنواع العنصر الجديدة . أما هذا الحال مختلف تماما مع وثائق XML

في حالة تغير امتداد الملف ليتم عرضة كوثيقة xml على المتصفح . هنا تظهر القدرة لتعريف العناصر الجديدة وعلاوة على ذلك ، يمكنها العمل على الوثيقة نفسها ، بحيث يكون الوصف عبارة عن وصف ذاتي -self describing .

دعنا نتخيل مثالا الشكل التالي كوثيقة HTML على هيئه جدول دورات تدريبية :

PHP Programming	**15-04-2003**	**Emad Adly**	**jimmy salh adel**
Java Programming	**8-03-2004**	**jimmy Adle**	**Devman karim**
ASP Programming	**12-5-2004**	**salh Mohammed**	**SeGa mrscript**

```
<table >
    <tr>
        <td> <i>PHP Programming</i> </td>
        <td> 15-04-2003</td>
        <td>  Emad Adly </td>
        <td> jimmy <br> salh <br> adle  </td>
    </tr>
```

(وهكذا لكل صف). . .

</table>

وبالرغم من إننا كبشر أو مبرمجين يمكن لأي شخص تخمين أن الاسم المذكور شفى هذا الجدول يدل عن انه اسم المعلم أو المحاضر للدورة المذكور بجوارها ، وليس من المعقول التعرف على PHP Programming كاسم المعلم ، أو من هم المشاركين في الدورة ، أما في البرمجة وفهم الحاسب لذلك بطريقة منهجية فان XML يثبت هذا في تعديل هذا الجدول هكذا :

```
<course>

    <name> PHP Programming </name>

    <date> 15-04-2003</date>

    <teacher> Emad Adly </teacher>

    <student> Salh  </student>

    <student> Jimmy </student>

    <student> Adle  </student>

</course>
```

في المثال السابق لـ XML هو يعتبر وثيقة XML قياسية يمكنك أن تعطى إي بيانات تعطي المعنى المناسب للوصف الخاص بهذه البيانات والتي تم تعريفة كوصف قياسي ذاتي لوثيقة XML . باختيارنا لـ Tags تعطى وصف واضح وصريح لما تحتوى من بيانات ،

فهكذا اصبح نص قياسي لذا يمكن أن يتحول بسهولة من جهاز إلى جهاز أو نظام إلى نظام ، لذا إي واحد يمكنه أن يفهم أن هذه الوثيقة تحمل شي من التوضيح إنها دورة Course .

والسؤال الذي يدور في ذهن الآن هو كيف لي أن أهيئ هذا ليظهر في شكل وصورة html على المتصفح ؟

نلاحظ أن **HTML** تحتوى على تعليمات لوصف النص على المتصفح فعلى سبيل المثال **PHP Programming** يشير إلى انه نص **italics** ، هنا نجد أن **XML** ليس عنده عدد ثابت من الـ **tags** مثلما يعمل **HTML** ولكنه قابل للامتداد مثل لغة **SGML** تسمح لمصمم الوثيقة بتعريف البيانات الخاصة ، **XML** جاءت لتلبيه النقص في هذه اللغات والحاجة لنشر المعلومات التي تتضمن شبكات المعلومات الكبيرة والنشر الورقي التقليدي واستعمالها على أنظمة غير تقليدية ، وكلما توسعه الإنترنت ستصبح اكثر وضوحا.

لم نتطرق في هذه السلسلة إلى كيفية استخراج البيانات وعرضها على المتصفح بطرق غير تقليدية أو بصورة **html** باستخدام لغة **XSLT**

وأيضا لم نتطرق إلى الوصول إلى البيانات في مستندات **XML** برمجيا بواسطة تقنيات **DOM** و **SAX**

ملحوظة : هذا الملحق منقول من موقع http://www.c4arab.com

للكاتب : أسماء المنقوش

الموسوعة العربية للكمبيوتر و الانترنت

خالص الشكر و التقدير للأسرة الموقع و كاتب الموضوع و نحتفظ بكافة الحقوق فى هذا الموضوع للكاتب و للموقع .

ملحق : هندسة البرمجيات

مقدمة:

لم يعد خافيا على أي منا أهمية البرمجيات Software في حياتنا اليومية سواء في البيت أو المصنع أو المستشفى أو ... الخ، فنحن نتعامل يوميا مع العديد من الأجهزة والمعدات التي تعتمد في عملها على البرمجيات ومن المهم لنا أن تعمل هذه الأجهزة وبرامجها بالشكل والكفاءة التي نتوقعها منها. لذا فإن هندسة البرمجيات أصبحت اليوم أكثر أهمية من أي وقت مضى.

المرجع:

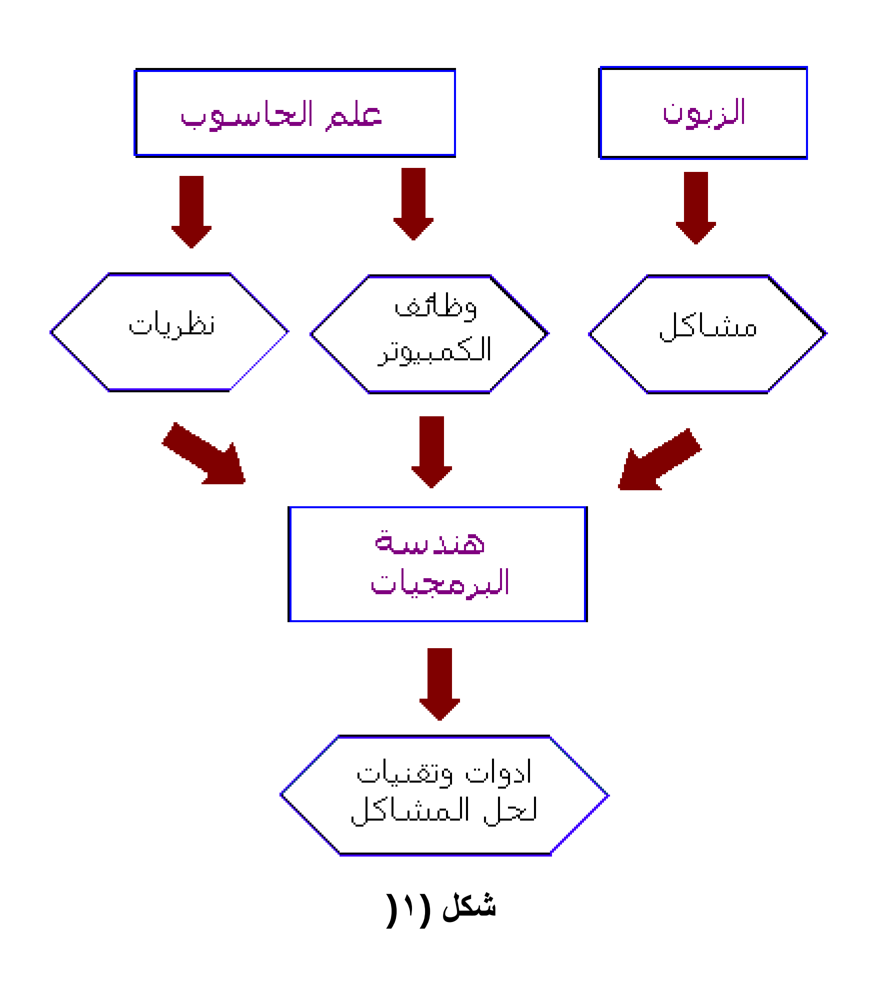
1- Shari Pfleeger, "Software Engineering - Theory and Practice", 2nd Edition

ما هي هندسة البرمجيات؟

لنفهم معا علاقة هندسة البرمجيات بعلوم الكومبيوتر، دعونا نأخذ هذا المثال عن علم الكيمياء واستخدامه في حل المشاكل التي نقابلها في حياتنا اليومية.
يهتم الكيميائي بدراسة المواد الكيميائية (تركيبها، تفاعلاتها، والنظريات التي تحكم سلوكها.)
بينما المهندس الكيميائي يستخدم النتائج التي توصل إليها الكيميائي لحل المشاكل التي يطلب منه إيجاد حل لها.
من وجهه نظر الكيميائي الكمياء هي موضوع الدراسة بحد ذاتها.
ومن وجهه نظر المهندس الكمئيائي الكيمياء هي أداة tool تستخدم لأيجاد الحلول لمشاكل عامة) وقد لا تكون هذه المشكلة ذات طبيعة كيميائية بحد ذاتها.(

وبنفس الفكرة يمكن النظر إلى علم الحوسبة computer science حيث يكون تركيزنا على الحواسيب ولغات البرمجة لدرستها وتطويرها في حد ذاتها.
أو يمكن النظر إليها والتعامل بها على أنها أدوات نستخدمها عند تصميم وتطوير حل لمشكلة ما تواجهنا أو الآخرين.

مهندس البرمجيات Software Engineer يعتبر أن الكمبيوتر هو أداة لحل المشاكل problem-solving tool.
وعليه أن يستخدم معلوماته حول الحاسوب وعلم الحوسبة للمساعدة في حل المشكلة التي يطلب منه إيجاد حل لها.



شكل (١)

ولكن ومن المهم أن نتذكر أن عملية كتابة البرامج تعد فن Art بقدر ما هي علم، لماذا؟

لأنه يمكن لأي شخص لديه معرفة كافية بأحد لغات برمجة الحاسوب hacker أن يكتب برنامج ليؤدي مهمة محددة، لكن الامر يتطلب مهارة ومعرفة مهندس برمجيات محترف لكتابة برنامج أكثر تناسقا ووضوحا ،وأسهل في الصيانة، ويقوم بالمهمة المطلوبة منه بفعالية ودقة أكبر.

أي أن، هندسة البرمجيات تعنى بتصميم وتطوير برامج ذات جودة عالية.

من يشارك في هذه العملية؟
المشاركون في عملية صناعة البرنامج، عادة ما يندرجون تحت ثلاث مجموعات:

- الزبون :Customer وهو الشركة (أو الشخص) الممولة لمشر وع تطوير البرنامج المطلوب
- المستخدم :User الشخص (أو مجموعة الاشخاص) الذي سوف يقوم فعلا باستعمال البرنامج، والتعامل معه مباشرة.
- المطور :Developer وهو الشركة (أو الشخص) الذي سوف يقوم بتطوير البرنامج لصالح الزبون.

الشكل التالي يظهر العلاقة بين الفئات الثلاثة السابقة

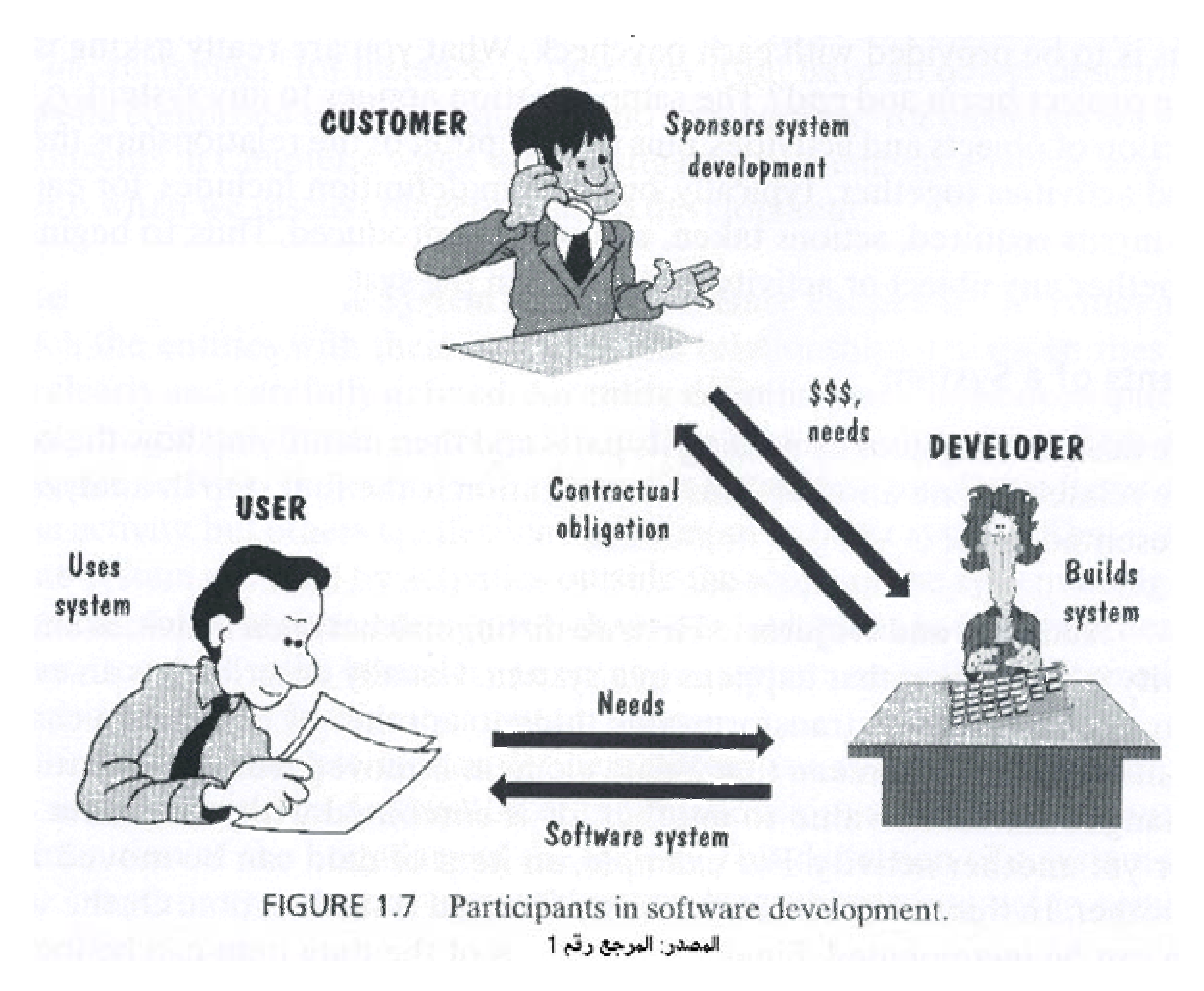


FIGURE 1.7 Participants in software development.

المصدر: المرجع رقم 1

شكل (٢)

مكونات النظام

مشاريعنا التي نطورها لن تعمل في الفراغ، فعليها أن تتفاعل مع مستخدمين، أجهزة ومعدات متنوعة، نظم تشغيل وبرامج وملفات وقواعد بيانات .... إلخ و ربما حتى أنظمة حواسيب آخرى. لهذا يجب تعريف حدود النظام ومكوناته جيدا .أي يجب تعريف ما الذي يشتمل عليه النظام وما الذي لا يشتمل عليه.

أي نظام هو عبارة عن مجموعة من الكائنات objects والنشاطات activities بالإضافة إلى وصف للعلاقات التي تربط تلك الكائنات والنشاطات معا. مع تعريف قائمة المدخلات المطلوبة والخطوات المتبعة والمخرجات الناتجة لكل نشاط.

أول خطوات تحليل المشكلة هو فهم ماهية المشكلة وتعريفها بوضوح، لذا علينا أولا أن نصف النظام بتحديد مكوناته والعلاقات التي تربط بين هذه المكونات.

.1النشاطات والكائنات: النشاط هو عملية تحدث بالنظام وعادة ما يوصف كحدث يتم من خلال حافز. النشاط يغير شئ ما إلى آخر بتغير خواصه (صفاته(

هذا التغير يمكن أن يعنى تحويل أحد عناصر البيانات من موقع إلى آخر، أو تعديل قيمته إلى قيمة مختلفة.

هذه العناصر تسمى كائنات objects وهي عادة ماتكون مرتبطة ببعضها البعض بشكل أو بأخر. مثلا الكائنات يمكن أن تكون مرتبة في مصفوفة أو سجل) قيد.(

وصف هذه الكائنات نوعها، النشاطات التي يمكن إجرائها عليها ... يجب وضعها بدقة هي ايضا.

2.العلاقات وحدود النظامRelationships and System Boundary

بعد تعريف الكائنات والنشاطات جيدا، يمكن أن نربط بين كل كائن والنشاطات المتعلقة به بدقة. تعريف الكائن يتضمن الموقع الذي سوف ينشأ به(نعض العناصر يمكن أن تكون موجودة بملف سبق انشاءه، والبعض قد يتم انشاءه خلال حدث ما(، والهدف من انشاءه(بعض الكائنات تستخدم من قبل نشاط واحد فقط والبعض يمكن أن يستعمل من قبل نظم آخرى كمدخلات , (Input لذا يمكن أن نعتبر أن لنظامنا حدود boundary بعض الكائنات بمكن أن تعبر هذه الحدود إلى داخل النظام، والبعض الآخر هي مخرجات من نظامنا ويمكن أن ترحل إلى نظم آخرى.

بهذا يمكن أن نعرف النظام A System على أنه تجمع من:
- **مجموعة من الكائنات.entities**
- **مجموعة من الانشطة.activities**
- **وصف للعلاقات بين الكائنات والانشطة.Relationship**
- **تعريف لحدود النظام.boundary**

كيف نبني نظام؟

إذا طلب منا عميل تطوير نظام (برنامج) له، لحل مشكلة معينة تواجهه في عمله. فمثلا يحتاج نظام حماية لشركته، أو نظام صرف آلي لبنك، أو ممكن أن يكون صاحب مكتبة أو متجر و يريد تغير نظام البيع و الشراء أو العرض ليتم بشكل آلي. علينا اتباع الخطوات التالية لبناء هذا النظام:
1. **عقد اجتماع مع العميل لتحديد متطلباته، هذه المتطلبات تشمل وصف النظام بجميع مكوناته التي شرحنا.**
2. **وضع تصميم عام للنظام يحقق المتطلبات التي حددها العميل، وعرضه على العميل ليوضح له الشكل الذي سيظهر عليه النظام عند الانتهاء، و ومراجعته معه لأخذ موافقته عليه.**
3. **بعد موافقة العميل على التصميم يتم العمل على وضع التصاميم التفصيلية لأجزاء المشروع.**
4. **كتابة البرنامج**
5. **اختباره، واعادة مراجعة المتطلبات التي وضعها العميل للتأكد من تحققها في البرنامج.**
6. **تسليم النظام إلى العميل.**
7. **بعد تسلم العميل للنظام قد تظهر بعض المشاكل أو الاخطاء التي لم تظهر خلال عملية الاختبار، والتي تجب على المطور اصلاحها فيما يعرف بصيانة النظام**

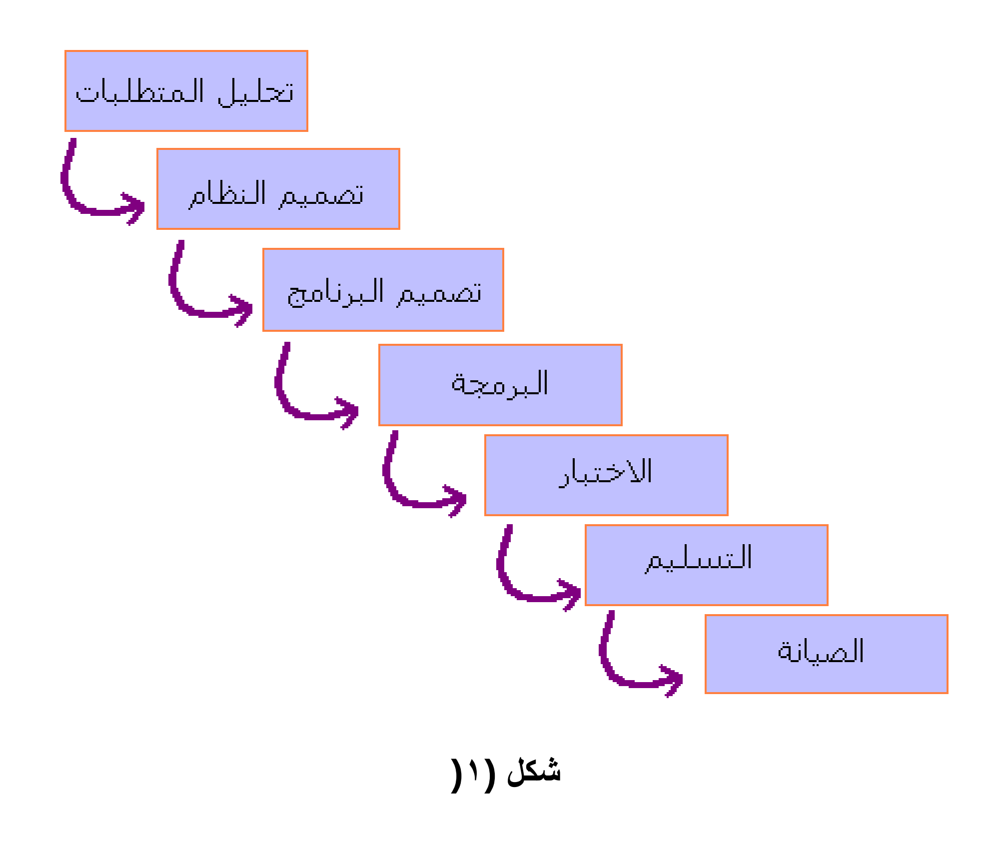
عملية بناء أي منتج تمر بعدة مراحل يطلق عليها عادة "دورة الحياةLife Cycle "، ومما تعلمنا في الدرس السابق فإن دروة حياة تطوير أي نظام برمجي Software development life cycle تتضمن المراحل التالية:
1. **تحديد وتعريف المتطلباتRequirements analysis and definition**
2. **تصميم النظامSystem design**
3. **تصميم البرنامجProgram design**
4. **كتابة البرنامج (تطويرهProgram implementation)**
5. **أختبار وحدات البرنامجUnit testing**
6. **أختبار النظامsystem testing**
7. **تسليم النظامsystem delivery**
8. **الصيانةmaintenance**

كل مرحلة من تلك المراحل تتضمن العديد من الخطوات أو النشاطات ولكل منها مدخلاتها ومخرجاتها وتأثرها على جودة المنتج النهائي) البرنامج.(
دورة حياة أي منتج تبدأ بأول خطوة وهي تحديد المتطلبات وتتدرج إلى باقي الخطوات كما هي مرتبة حتى الوصول إلى آخر خطوة وهي تسليم البرنامج وصيانته (إن دعت الحاجة)، إلا أن التجارب العملية تظهر أن هذا ليس ضروريا وأن دورة حياة تطوير البرامج قد تأخذ أشكال (أو أنماط) مختلفة. وفي هذا الدرس سوف نتعرف إلى هذه الأنماط

أنماط دورة الحياة:Lifecycle Models

النموذج الانحداري Waterfall Model

في هذا النموذج تسير دورة الحياة بشكل تدريجي بدأ من الخطوة (١) وحتى الخطوة (٨)، وكما يظهر بالشكل (١) فإن كل مرحلة تبدأ بعد الأنتهاء من المرحلة التي تسبقها مباشرة.



شكل (١)

يتميز النموذج الانحداري بالبساطة، ولذا فإنه يسّهل على المطور توضيح كيفية سير العمل بالمشروع للعميل (الذي عادة لا يعرف الكثير عن صنع البرمجيات) والمراحل المتبقية من العمل. وقد كان هذا النموذج أساس عمل كثير من المؤسسات لفترة طويلة مثل وزارة الدفاع الامريكية، واستنبط منه العديد من النماذج الاكثر تعقيدا.

إلا أن لهذا النموذج العديد من العيوب، أهمها أنه لا يعكس الطريقة التي يعمل بها المطورون في الواقع. فباستثناء المشاريع الصغيرة والبسيطة (أي أنها مفهومة بشكل جيد للمطور) فإن البرمجيات عادة ما تنتج بعد قدر هائل من التكرار والاعادة. في حين أن هذا النموذج يفترض أن يكون الحل واضح ومفهوم وسبق تحليله بالكامل قبل مباشرة مرحلة التصميم وهو أمر يكاد يكون شبه مستحيل مع الانظمة الضخمة. وحتى إن كان ممكن فإنه يأخذ وقت طويل جدا (ربما سنوات!)

باختصار،النموذج الانحداري سهل الفهم و بسيط في إدارته. لكن مميزاته تبدأ في التداعي بمجرد أن يزداد تعقيد المشروع.

التطوير على مراحل Phased Development

حسب النموذج الانحداري فإنه يجب على المطورين إنهاء مرحلة تحليل المشروع بشكل تام قبل البدأ في التصميم، وكما وضحنا فإن هذه المرحلة قد تتطلب وقت طويل في بعض المشاريع وقد تمر عدة سنوات قبل أن يرى البرنامج النهائي النور، ولكن هل يمكن لسوق العمل الانتظار كل هذا الوقت؟!

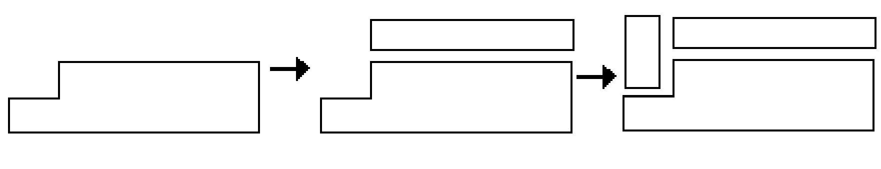
الاجابة بالطبع لا.

لذا كان لابد من ايجاد طرق آخرى لتقليل زمن تطوير المشروع Cycle time. أحد هذه الطرق هي التطوير على مراحل Phased Development حيث يتم تطوير النظام على عدة مراحل، بتقديم إصدار من البرنامج به بعض الوظائف للعميل والعمل على تطوير الاصدار الاحق الذي سوف يقدم له بقية الوظائف.

يوجد عدة طرق يمكن بها تنظيم عملية تطوير إصدارات البرنامج، ومن اشهرها:

▪ النموذج التزايدى Incremental model

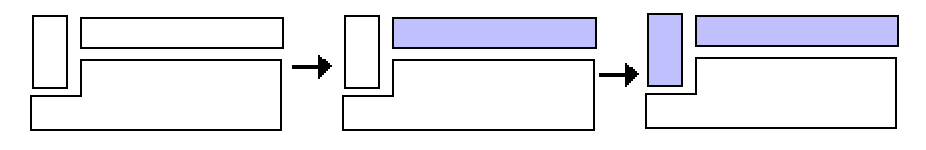
حيث يتم تقسيم النظام المطلوب تطويره إلى عدة اجزاء حسب الوظائف التي يعتين عليه القيام بها، يبدأ أول إصدار بأحد تلك الاجزاء ومع الوقت يتم إضافة المزيد من الاجزاء (الوظائف) حتى يتم الانتهاء من تطوير النظام بشكل تام وحسب متطلبات العميل.



Incremental Development

▪ النموذج التكرارى Iterative model

هذه المرة يتم تسليم برنامج بكامل الوظائف من أول مرة، ولكن يتم تعديل وتغيير بعض تلك الوظائف مع كل إصدار من البرنامج.



Iterative Development

من مميزات هذا الأسلوب أنه يمكن المطورين من الحصول على ملاحظات وتقييم الزبون مبكرا و بصورة منتظمة، ورصد الصعوبات المحتملة قبل التمادي بعيدا في عمليات التطوير.كم أنه يمّكن من اكتشاف مدى حجم و تعقيد العمل مبكرا.

النموذج اللولبي Spiral Model

وهو شبيه لدرجة كبيرة إلى النموذج التزايدي والتكراري، ولكن فيه يتم دمج فعاليات التطوير مع إدارة المخاطر risk من إجل التحكم بها وتقليلها.

يبدأ النموذج اللولبي بمتطلبات العميل مع خطة العمل المبدئية (الميزانية، قيود النظام، والبدائل المتاحة). ثم يتقدم خطوة إلى الامام بتقدير المخاطر وتمثيل البدائل المتاحة قبل تقديم ما يعرف بـ "وثيقة العمليات " Concept of Operations التي تصف وبشكل عام (بدون الدخول في التفاصيل) كيف يجب على النظام أن يعمل. بعدها يتم تحديد وتدقيق المتطلبات للتأكد من أنها تامة ودقيقة إلى أقصى حد ممكن.

بذلك تكون وثيقة العمليات هي المنتج من الطور الأول، و المتطلبات هي المنتج الاساسي من الطور الثاني. وفي الطور الثالث تتم عملية التصميم، أما الاختبار فيتم خلال الطور الرابع.

في كل طور أو مرحلة يساعد تحليل المخاطر على تقدير البدائل المختلفة في ضوء متطلبات وقيود النظام، وتساعد النمذجة على التحقق من ملائمة أي بديل قبل أعتماده.

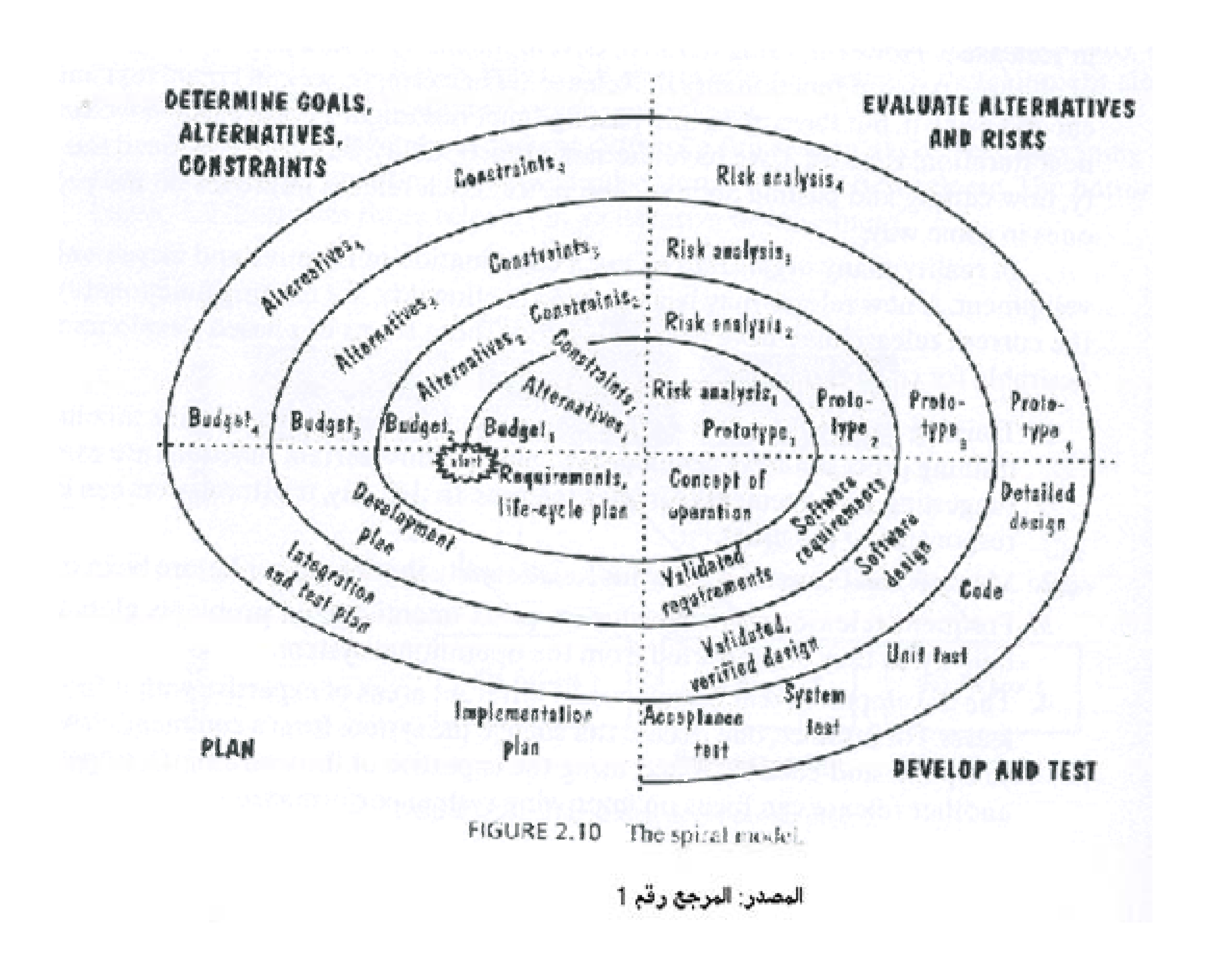


FIGURE 2.10 The spiral model.

المصدر: المرجع رقم 1

سوف نبدأ في دراسة أول (ولعلها أهم) خطوة في تطوير البرامج وهي تحديد متطلبات النظام Capturing the requirements.

الهدف من تحديد المتطلبات هو فهم ما يتوقعه العميل والمستخدم من النظام (ما الذي يمكن للنظام أداؤه وما لا يمكنه أداؤه).فقد يكون النظام المطلوب تصميمه بديل لنظام أو لطريقة مستخدمة لأداء مهمة محددة، أو ممكن أن يكون نظام جديد يقدم خدمة جديدة لم يسبق تقديمها من قبل. فلكل نظام برمجي وظيفة معينة، تحدد بما يمكن له أن يقوم به من أجل أداء تلك الوظيفة.

المتطلبات :هي تعريف لشكل النظام أو وصف لما يستطيع هذا النظام أن يقوم به لأداء وظيفته التي سيصمم من أجلها.

خطوات تحديد المتطلبات :

أولا: الاجتماع مع العميل للتعرف على المتطلبات:

وهذه خطوة هامة جدا إذ أن بقية الخطوات التالية تعتمد عليها بشكل أساسي. لذا يجب علينا أن نستخدم كافة التقنيات المتاحة لنكتشف ما الذي يطلبه العميل والمستخدم، نبدأ بفهم وتحليل المشكلة التي تواجه المستخدم بكل أبعادها، نتعرف على العمليات والمصادر التي تتضمنها المشكلة والعلاقات التي تربطها معا و نحدد حدود النظام. وهذا يمكن أن يتم من خلال:

- طرح الأسئلة على العميل، ومن المفيد أحيانا أن نطرح نفس السؤال ولكن بأسلوب مختلف أكثر من مرة فهذا يساعدنا على التأكد من أننا نفهم ما يقصده العميل بالتحديد.
- عرض نظم مشابه للنظام المطلوب سبق تصميمها من قبل.
- تصميم وعرض نماذج لأجزاء من النظام المطلوب أو للنظام بالكامل.

تقسم المتطلبات إلى عدة عناصر تشمل:

- البيئة المحيطة بالنظام Physical Environment
- وجهات الاستخدام Interfaces
- المستخدمين وإمكاناتهم Users and human factors
- وظائف النظام Functionality
- التوثيق Documentation
- البيانات Data
- المصادر Resources
- الأمن Security
- ضمان الجودة Quality Assurance

ويجب التأكد من أن نناقش جميع هذه العناصر

ثانيا: تسجيل هذه المتطلبات في وثائق أو قاعدة بيانات، وعرضها على العميل ليوافق عليها باعتبار أنها ما يطلبه بالفعل

المتطلبات لا تصف فقط تدفق البيانات والمعلومات من وإلى النظام، وأما تصف كذلك القيود المفروضة على عمل النظام. وبذلك فإن عملية تحديد المتطلبات تخدم ثلاثة أغراض:

- أولا تمكن المطورين من شرح فهمهم للطريقة التي يود المستخدمين أن يعمل بها النظام.
- ثانيا توضح للمصممين ماهية الوظائف والخصائص التي سيمتاز بها النظام ,
- وثالثا: توضح المتطلبات لفريق الاختبار ما الذي يجب إثباته لإقناع الزبون أن النظام الذي تم تطويره هو ما سبق أن طلبه بالضبط .

لذلك ولضمان أن كلا من المطورين والزبون متفاهمون تماما على ما يجب القيام به، فإن المتطلبات المسجلة حتى هذه الخطوة يجب أن تكون لها الصفات التالية:

1. أن تكون صحيحة Correct وخالية من الأخطاء.
2. أن تكون ثابتة consistent بمعنى أن لا يكون هناك أي تعارض بين متطلب وآخر.
3. أن تكون تامة Complete يجب أن يتم ذكر جميع الحالات المحتملة للنظام، المدخلات، المخرجات المتوقعة منه، ...الخ .
4. أن تكون واقعية realistic بمعنى أن تكون قابلة للتطبيق في الواقع.
5. أن تكون متعلقة بأمور ضرورة للعميل، ويتطلبها النظام.
6. أن يكون من الممكن التحقق منها verifiable
7. أن تكون قابلة للتتبع traceable

يطلق على هذه الوثائق "وثائق تعريف المتطلبات Requirement Definition Document "

ثالثا: إعادة تسجيل المتطلبات بشكل رياضي mathematical ليقوم المصممون بتحويل تلك المتطلبات إلى تصميم جيد للنظام في مرحلة التصميم.

لسنوات عديدة كان يتم الاكتفاء بوثيقة تعريف المتطلبات (التي تحدثنا عنها قبل قليل) والتي تكتب باستعمال اللغة الطبيعية) لغة البشر) لوصف وتسجيل متطلبات النظم بحيث يمكن للعميل أن يفهم كل كلمة موجودة بها، إلا أن ذلك يسبب العديد من المشاكل والتي يعود سببها في أغلب الأحيان إلى سوء تفسير بعض التعبيرات للمستخدمين من قبل

المصمم أو العكس، فعلى سبيل المثال قد يطلق المستخدم على النظام التعبير (متوقف عن العمل) إذا كان النظام مشغول بعملية تسجيل احتياطي backup باعتبار أن لا يستجيب لأوامر المستخدم في هذه الحالة، بينما يعتبر المصمم أن النظام في هذه الحالة (مستمر في العمل) لأنه يقوم بمهمة أساسية!

لذا فأن الاعتماد على اللغة البشرية بشكل تام قد يؤدي إلى أخطاء كثيرة عند تصميم النظام، وينتج عنها نظام لا يقبله العميل لأنه لا يلبي متطلباته التي حددها من قبل، لذلك يتم كتابة نوع ثاني من الوثائق تسمى "وثائق مواصفات المتطلبات Requirement specification Document" وهي تكتب باستعمال وسائل وطرق خاصة ابتكرها مهندسو البرمجيات لكتابة المتطلبات باسلوب تقني بحت. منها على سبيل المثال: لغة النمذجة الموحدة UML Unified Modeling Languageو هي لغة نمذجة رسومية تقدم لنا صيغة لوصف العناصر الرئيسية للنظم البرمجية.

الشكل التالي يعرض مثال على استعمالUML

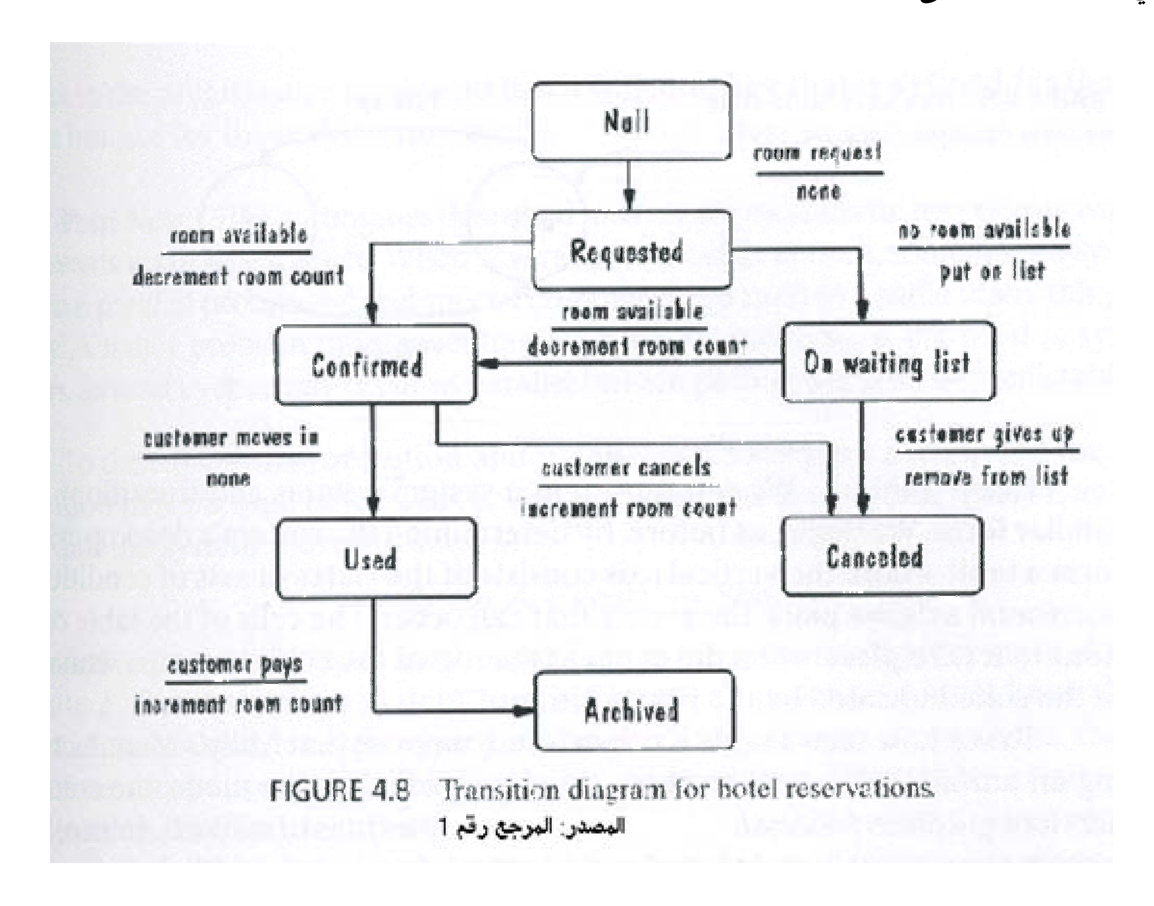


FIGURE 4.8 Transition diagram for hotel reservations
المصدر: المرجع رقم 1

رابعا: التثبت والتحقق من المتطلبات التي تم تسجيليها في كلا من وثيقة تعريف المتطلبات (والتي تقدم للعميل) ووثيقة مواصفات المتطلبات (والتي تقدم للمصمم) (للتأكد من صحتهما وشموليتهما وأن كلا منهما لا تعارض الثانية في أي نقطة، وإلا فإن النتيجة سوف تكون نظام لا يلبي طلبات العميل.!

نكمل مع خطوات بناء النظام، وهذه المرة سوف نتحدث عن خطوة "تصميم النظام "

ما هو التصميم؟

التصميم هو عملية إبداعية لإيجاد حل لمشكلة، كما تطلق عادة كلمة تصميم على وصف هذا الحل.
حيث نستفيد من المتطلبات التي حددنها في الخطوة السابقة في التعرف على المشكلة، ثم نبدأ في التفكير في الحل الذي يفي بجميع الشروط والمواصفات التي تحددها المتطلبات، وغالبا ما يمكن إيجاد عدد غير محدود من الحلول يمكن لنا أن نختار أحدها و الذي نجده الأنسب من بينها.

عند الانتهاء من خطوة تحديد المتطلبات، فإننا ننتهي بوثيقتين (كما ذكرنا في الدرس السابق) الأولى هي (وثيقة تعريف المتطلبات) ويتم تقديمها للعميل والثانية (وثيقة مواصفات المتطلبات) ويتم تقديمها للمصمم.

ودور المصمم هو تحويل هذه الوثائق إلى نظام يرضي العميل (يلبي احتياجاته)، وفي نفس الوقت يرضي المطور (يمكن تطبيقه.)
لذا فإن عملية التصميم في عملية تكرارية **iterative** من خطوتين:

أولا: يتم إنتاج التصميم التصوري **conceptual design** والذي يوضح للعميل ما الذي سيقوم به النظام بالتحديد . وفي حال موافقة العميل على هذا النظام، يتم الانتقال للخطوة التالية.

ثانيا: تحويل التصميم التصوري إلى وثيقة بها تفاصيل أكثر عن التصميم يطلق عليها اسم التصميم التقني **technical design** والذي يجب أن يظهر للمطور ما هي المعدات والبرمجيات اللازمة لبناء النظام.

أحيانا يتطلب الأمر للعودة إلى الخطوة الأولى (التصميم التصوري) والتعديل عليه، لذا فأنها عملية تكرارية حتى الوصول إلى التصميم الذي يرضي العميل ويمكن تطبيقه على أرض الواقع في ظل الإمكانيات المتاحة للمطورين.

التصميم التصوري:conceptual design

يركز هذا التصميم على وظائف النظام **functions** ويكتب بلغة يمكن للعميل أن يفهمها (لغة البشر) ليجيب عن أسئلة العميل حول ماذا (WHAT) يعمل النظام. ويجب أن يكون خالي تماما من أي تفاصيل برمجية أو فنية. والاهم أن يحقق كل المتطلبات التي تم تحديدها سابقا.

التصميم التقنيtechnical design

هذا التصميم سوف يتم تقديمه إلى مطوري النظام ليقوموا هم بتحويله إلى النظام المطلوب، لذا يجب أن يقدم هذا التصميم إجابة شافية لأسئلة المطور عن كيفية (HOW) تطوير النظام. ولمنع إلى تضارب في المفاهيم فإن هذا التصميم عادة ما يكتب باستعمال تعبيرات وأساليب تقنية.

كتابة البرنامج واختباره

هذا الدرس لن يعلمك لغة برمجة لتكتب بها البرامج، ولكن الهدف منه التعرف على:

- القواعد الصحيحة لكتابة البرامج
- خطة الاختبار وأنواع الاختبارات

الجزء الأول :كتابة البرامج:

بعد وضع التصميم للنظام واختيار لغة البرمجة المناسبة، تبدأ الخطوة التي سوف تنقل التصميم المكتوب على الورق إلى واقع. خلال هذا الدرس سوف نناقش أهم القواعد التي على المبرمج إتباعها أثناء كتابة برامجه. ولكن قبل ذلك لنجيب على هذا السؤال الذي لا شك أنه ورد على ذهنك الآن

س: لماذا علينا إتباع هذه القواعد؟

ج :إذا كنت تعمل منفردا في كتابة برامجك، فإن إتباعك لقواعد وأساليب قياسية في البرمجة سوف تساعدك على تنظيم أفكارك لتجنب الوقوع في الأخطاء. كما أنها ستساعدك على اكتشاف أي أخطاء قد تحدث بسرعة وبسهولة.

أم إذا كنت تعمل ضمن فريق برمجي، فإن إتباع القواعد والأساليب القياسية في كتابة أجزاء البرامج التي يطلب منك كتابتها، سوف تساعدك وبقية الفريق من تنسيق أعمالكم وتنظيمها، كما أنها ستقلل من عدد الأخطاء في البرنامج وتساعد على اكتشاف ما يقع منها في اسرع وقت ممكن.

تفرض الكثير من شركات البرمجة على مبرمجيها إتباع قواعد قياسية في كتابة برامجهم، وذلك لضمان التكامل في جميع البرامج، كما أن بعض الشركات تعين فرق لاختبار البرامج، غير الفريق الذي قام بالبرمجة ولذلك يجب أن يكون الكود البرمجي مكتوب بطريقة واضحة لجميع من يقرأه، وليس لمن قام بكتابته فقط.

بعض قواعد البرمجةProgramming Guidelines

- هياكل التحكمControl Structures

يقصد بها تلك الهياكل التي تتحكم في مسار عمل البرنامج (مثل Goto، if- else) ، وأثناء كتابة هذه الهياكل علنا أن نحاول أن نجعلها واضحة وسهلة التتبع، وخالية من القفزات الواسعة قدر الإمكان. انظر لهذا المثال:

```
benefit = minimum;
     if (age < 75) goto A;
    benefit = maximum;
    goto C;
    if (age < 65) goto B;
    if (age < 55) goto C;
A:   if (age < 65) goto B;
    benefit = benefit * 1.5 + bonus;
    goto C;
B:   if (age < 55) goto C;
    benefit = benefit * 1.5;
C:  next statement
```

نفس الكود يمكن كتابته على هذا النحو:

```
if (age < 55)  benefit = minimum;
   else if (age < 65)  benefit = minimum + bonus;
   else if (age < 75)  benefit = minimum * 1.5 +bonus;
   else  benefit = maximum;
```

- **عالم البرمجة هناك قاعدة تقول أن العمومية ميزةgenerality is a virtue ، لذلك حاول دائما أن تجعل شفراتك البرمجة عامة، لتتمكن من إعادة استعمالها في بقية برامجك بأقل قدر ممكن من التعديل، ولكن حاذر من التمادي في ذلك!**
- **لا تستخدم أبدا أسماء لا معنى لها لمتغيرات أو بارمترات برنامجك (ينصح بمراجعة هذا الدرس" التسمية في البرنامج، درس لابد من أن يقرأه كل مبرمج!)"**
- **"أريد برنامجا سريعا" وكلنا نريد ذلك، ولكن ما هو الثمن؟!**

عندما تفكر في جعل برنامجك أسرع ما يمكن، عليك أن تفكر كذلك في الثمن الذي ستدفعه مقابل ذلك:

١. **البرنامج السريع قد يتطلب منك كتابة كود معقد يتطلب منك (ومن فريق العمل) المزيد من الوقت والجهد في كتابته.**
٢. **الوقت الذي تحتاجه عملية اختبار البرنامج المعقد في مختلف حالته.**
٣. **الوقت والجهد الذي تحتاجه لتعديل هذا الكود أو لتطويره.**

زمن تنفيذ البرنامج ما هو إلا جزء من معادلة كبيرة لحساب تكلفة البرنامج، لذلك عليك أن تعادل بين السرعة، والجودة، واحتياجات الزبون. ولا تضحي بالبساطة والوضوح من أجل السرعة.

- **التوثيق: لا تهمل أبدا توثيق برنامجك، ما سُمي الإنسان إنسانا إلا لنسيانه.**

الجزء الثاني :اختبار البرامج:

وصلنا الآن إلى آخر مرحلة في تطوير النظام، وهي اختبار البرنامج للتأكد من أنه يعمل على النحو الذي يتوقعه الزبون.

قبل تسليم النظام النهائي إلى الزبون تجرى عليه الكثير من الاختبارات، بعضها يعتمد على ما الذي يتم اختباره مثلا:

)أحد مكونات البرنامج - مجموعة من المكونات - جزء من النظام - النظام بالكامل(

والبعض الأخر يعتمد على ما الذي نريد معرفته من هذه الاختبارات، مثلا:

- **هل يعمل النظام وفقا لما ورد في المتطلبات؟**
- **هل يعمل النظام وفقا لما ورد في التصميم؟**
- **هل يعمل النظام كما يتوقعه الزبون منه؟**

مراحل الاختبار:

عند العمل على اختبار نظام من الحجم الكبير، فإن عملية الاختبار تتم على عدة مراحل موجزها في ما يلي:

١. **اختبار المكون Module Testing أوcomponent Testing**

أول مراحل اختبار النظم، هي اختبار كل مكون على حدى بمعزل عن بقية مكونات النظام، للتأكد من عمله على النحو المتوقع منه. باختبار المعلومات المتحصل عليها (output) منه بعد إمداده بالبيانات اللازمة له.(input)

٢. **اختبار التكاملIntegration Testing**

بعد اختبار كل مكونات النظام والتأكد من سلامة تصميمها، يجب أن نتأكد من أنها ستعمل معا بشكل صحيح وأنه لا يوجد تضارب بين بعضها البعض بحيث أن المعلومات المنتقلة بين هذه المكونات تصل بالهيئة المتوقعة لها. وهذا هو الهدف من اختبار التكامل.

٣. اختبار الوظيفةFunction Testing

ويقصد به اختبار النظام بعد تجميع كل مكوناته للتأكد من أنه يؤدي الوظيفة التي يتعين عليه القيام بها، والموضحة في وثائق متطلبات النظام .عندما يجتاز النظام هذا الاختبار يمكننا اعتبار هذا النظام على أنه نظام عامل **Functioning System**

٤. اختبار الأداء Performance Testing

في هذه الخطوة يتم اختبار أداء البرنامج في بيئة عمل الزبون للتأكد من أن النظام متوافق مع بقية المتطلبات . عند اجتياز النظام لهذا الاختبار يتم التصديق على النظام **validated system** وبهذا فإننا نعتبر أن النظام أصبح جاهز حسب مفهومنا لما طلبه الزبون.

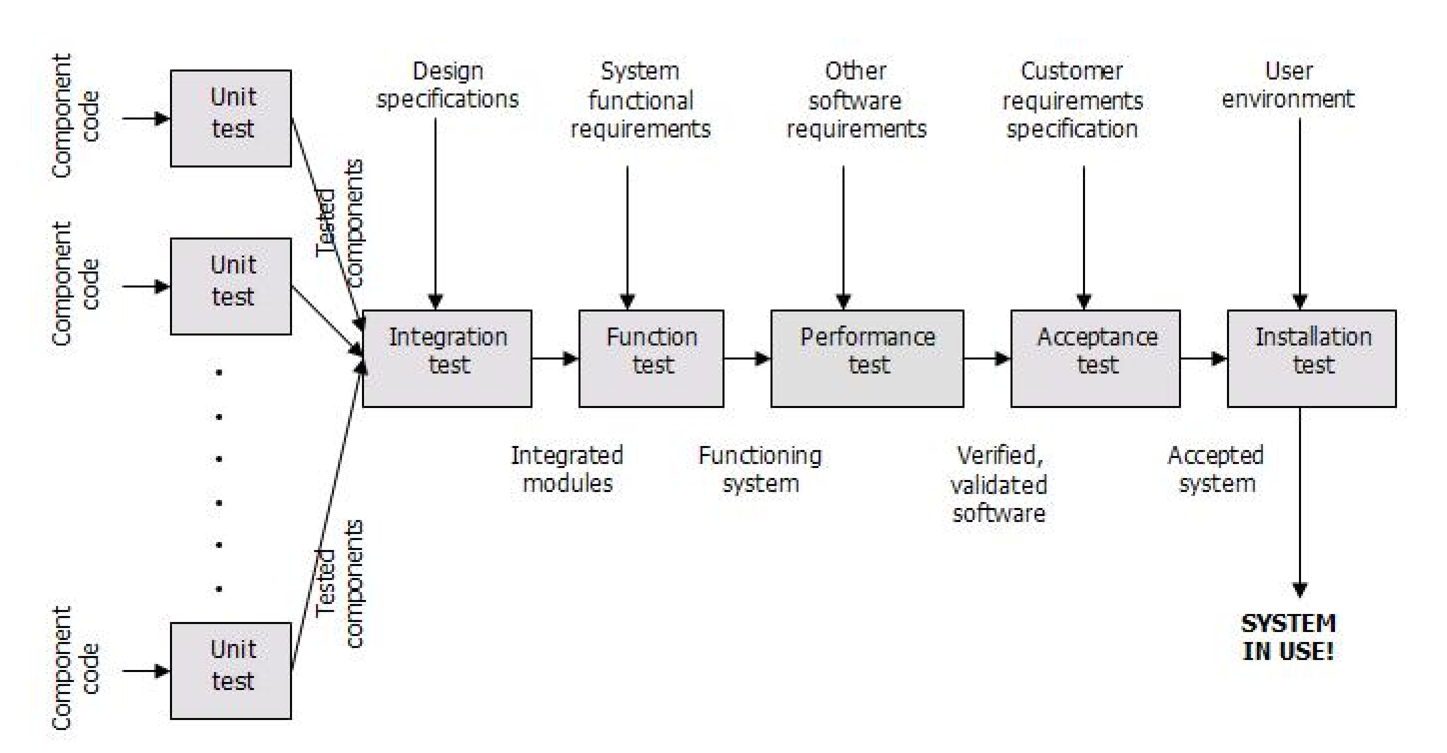
٥. اختبار القبولAcceptance Test

يتم إجراء هذا الاختبار للتأكد من أن النظام المحقق موافق لما توقعه الزبون، وبعدها يعد النظام مقبول عند المستخدم والزبون **Accepted system**

٦. اختبار التثبيتInstallation Test

الاختبار الأخير يتم فيه تثبيت النظام في بيئة العمل الخاصة به والتأكد من أنه يعمل كما هو مطلوب منه.

الشكل التالي يوضح خطوات تطبيق عملية اختبار النظام، والتي يحسن تطبيقها على اي نظام مهما كان حجمه للتأكد من أنه سيؤدي المهمة المطلوبة منه.



المراجع

Sams Teach Yourself Macromedia® Flash™ 8 in 24 Hours
By Phillip Kerman
Essential ActionScript 2.0
By Colin Moock
Object-Oriented Programming with ActionScript 2.0
By Jeff Tapper, James Talbot, Robin Haffner
Macromedia Flash MX 2004 Game Programming
by Craig S. Murray and Justin Everett-Church
Flash™ MX 2004 ActionScript Bible
Robert Reinhardt and Joey Lott
Object-Oriented
ActionScript for Flash 8
Peter Elst and Todd Yard
with Sas Jacobs and William Drol
Macromedia® Flash® 8 ActionScript: Training from the Source
By Jobe Makar, Danny Patterson

Learning ActionScript 2.0 for Macromedia® Flash® 8
Publisher: **Macromedia**
http://www.c4arab.com
http://xml.apache.org/
www.sadaagroup.com
http://www.roro44.com/lessons/lessons-59-530-0.html
ActionScript 3.0 Cookbook
By Joey Lott, Darron Schall, Keith Peters

اعتمدت فى شرح برنامج الاكسيس و لغة Sql و بعض امثلة فى Aspx على موقع
WWW.AS4WEB.COM
و مواقع اخرى متفرقة WWW.W3SCHOOLS.COM
WWW.ASPX.NET

السيرة الذاتية للمؤلف

Michael Nabil Akhnokh

Mobile : 0103546609

Michaelnail2006@yahoo.com

Education

- Faculty of Commerce, Ain Shams University
BA of Commerce
Major: Accounting
Graduation Year: 2001
- BBSA Nasr City, EG
Basic Business Skills Acquisition Program Sponsored by Future Generation Foundation
During the period from 13/1/2002 To 25/4/2002
- IT Professional Training Program provided by Ministry of communication and information Technology in cooperation with IBM Egypt
Specialty track: Solution Developer and programmer
During the period from 29/6/2003 To 15/1/2004

Qualifications and Skills

Computer Skills:

- **MCP (Microsoft Certified Professional) in:**
  Developing Web Applications with Microsoft Visual C#. NET (Exam 70 – 315), (January 2004)
  - **Operating Systems (Windows 98, Windows 2000 server, Linux)**
  - **Skillfully using all Office Applications (Access, Word, PowerPoint, Excel)**
  - **A background knowledge about Network**

- Programming Languages: (C, C++, C#, Java, Visual Basic 6.0)
-Web developing: (HTML, ASP, ASP.NET, PHP , JavaScript, VBscript)
-Web design: (FrontPage 2003, Photoshop 7, Switch MX,

Flash MX2004, Action Script 2 & 3)

- **Courses attended about Database:**

Database Methodology and Design

SQL Server 2000 Programming

MySQL 4.0

Oracle 9i DBA Fundamentals I

Other Courses

- Project Research, Development and Presentations.

- Business Concepts.

Experience and Projects:

---

1) Information Project Group(IPG) Company

Position: Working on a variety of projects in the field of distance learning.

Environment: FlashMx2004 and Action Script 2.0

- Please check(www.ipgegypt.com/afdl.htm)

2**) Worked as a web site developer**

(Please check:**www.ramegypt.cjb.net**, **www.faculity4eg.cjb.net** , **www.linuxeg.cjb.net**)

Hobbies

Chess, Computer.
